OpenStack最终用户文档（简体中文版）

Colin Lee (李广慧)

Published with GitBook



目錄

OpenStack最终用户文档（简体中文版）

摘要

OpenStack是一个开源的云计算平台，可以用在公有云和私有云上。若干互相关联的项目结合成了这样一整套的云基础设施解决方案。本文档向OpenStack最终用户介绍了如何用OpenStack dashboard和OpenStack命令行工具在OpenStack云上创建和管理资源。

本文档涵盖了OpenStack Liberty，OpenStack Kilo和OpenStack Juno这三个版本。

目录

OpenStack 控制台

云的最终用户可以用OpenStack控制台来管理自己的资源，这些资源的限制由管理员设定。读者可以修改本章的案例，来创建不同类型和配置的服务器实例。

登录控制台

控制台运行在具有 `nova-dashboard` 这一服务器角色的节点上。

1. 向这个云服务的管理员咨询能登录控制台的主机名，IP地址，账号和密码。
2. 打开一个启用了JavaScript和Cookies的浏览器。

   > 注意： 如果要使用VNC客户端，你的浏览器需要支持HTML5 Canvas和HTML5 WebSockets。VNC浏览器基于noVNC。详情请查阅noVNC: HTML5 VNC Client。想查阅支持的浏览器列表，参阅支持浏览器。

3. 在地址栏里，输入控制台的主机名或者IP地址。例如：

   ```
   https://ipaddressorhostname/
   ```

4. 在登录界面上，输入账号和密码，然后单击 `Sign In` 。
   窗口的最上边便会显示你的用户名。你还可以在这里找到设置和退出按钮。
   登录之后显示哪些选项卡取决于访问权限，用户角色，还取决于你用什么用户登录。
   - 如果你是以最终用户身份登录的，*Project*选项卡和*Identity*选项卡可用。
   - 如果你是以管理员身份登录的，*Project*选项卡，*Admin*选项卡和*Identity*选项卡可用。

OpenStack控制台 - *Project*选项卡

Project是云服务中的组织单位，也被成为**tenant**或者**account**。每一个用户都是一个或多个Project中的成员。用户创建和管理服务器实例，都是在Project里完成的。

在Project选项卡中，用户可以查看和管理某个project的资源情况，包括实例和镜像。在**CURRENT PROJECT**可以选择想看的project。

在*Project*选项卡下，可以查看如下选项卡。

*Compute*选项卡

- *Overview*：查看Project的概览。
- *Instances*：对服务器实的各种操作，包括查看，启动，关闭，暂停，重启和创建快照等，也可以在此处通过VNC连接服务器。
- *Volumes*：包含如下两个选项卡：

- Volumes: 查看，创建，编辑和删除Volume。
- Volume Snapshots: 查看，创建，编辑和删除Volume快照。
- *Images*: 查看Project用户创建的镜像和实例快照，以及所有公开可用的镜像。创建，编辑，删除镜像，以及通过镜像或快照启动实例。
- *Access & Security*: 包含如下四个选项卡：
    - Security Groups: 查看，创建，编辑和删除安全组和安全组规则。
    - Key Pairs: View, create, edit, import, and delete key pairs. 查看，创建，编辑，导入以及删除公私钥对。
    - Floating IPs: Allocate an IP address to or release it from a project. 向project分配IP；从project处回收IP。
    - API Access: 查看API端点。

***Network*选项卡**

- *Network Topology*：查看网络拓扑结构。
- *Networks*：创建和管理公司网络。
- *Routers*：查看和管理路由。

***Object Store*选项卡**

- *Containers*：创建和管理容器和对象。

***Orchestration*选项卡**

- *Stacks*: 使用REST API来编排管理多个复合云应用。
- *Resource Types*: 列出所有HOT模板支持的资源种类。

OpenStack控制台 - *Admin*选项卡

管理员用户可以通过*Admin*选项卡来查看使用量，管理实例，卷，flavors，镜像，服务和配额等。

在*Admin*选项卡下，可以查看如下选项卡。

***System*选项卡**

- Overview：查看概况
- Resource Usage：查看如下两个选项卡
    - Usage Report：查看用量报告。
    - Stats：查看所有资源的统计信息。

- Hypervisors：查看虚拟机管理器的概况。
- Host Aggregates：查看，创建和编辑主机聚合（？）。查看可用的域。
- Instance：查看，暂停，回复，停用，迁移，软/硬重启，删除其他某些project下的用户的实例。在这个选项卡中还可以查看日志，以及通过VNC连接服务器实例。
- Volumes：使用如下两个选项卡。
  - Volumes：查看，创建，管理和删除卷。
  - Volume Types：查看，创建，管理和删除卷类别。
  - Volume Snapshots：查看，管理和删除卷快照。
- Flavors：查看，创建，编辑，删除预先编排的云主机类型，或者查看额外的介绍。不同的云主机类型有不同的实例大小。
- Images：对定制的镜像做如下操作：查看，创建，删除，编辑属性。
- Networks：对网络做如下操作：查看，创建，删除，编辑属性。
- Routers：对路由做如下操作：查看，创建，删除，编辑属性。
- Defaults：查看默认的配额值。配额是在OpenStack Compute中硬性安排的，定义了最大的可用实例，以及资源的数量。
- Metadata Definitions：导入命名空间，查看元数据信息。
- System Information：该选项卡有如下子选项卡。
  - Services：查看服务列表。
  - Compute Services：查看计算服务列表。
  - Block Storage Services：查看块存储服务列表。
  - Network Agents：: 查看网络？View the network agents.
  - Orchestration Services：查看编排服务列表。

OpenStack控制台 - *Identity*选项卡

- Projects: 对Project做如下操作：查看，创建，分配用户，移除用户，删除。
- Users: 查看，创建，启用，停用，删除用户。

OpenStack控制台 - *Setting*选项卡

登录进任意账户，在左边便可以看见*Setting*选项卡。

- User Settings: 查看和管理控制台选项。
- Change Password: 修改用户密码。

上传和管理镜像

虚拟机镜像（本文简称镜像），是一个包含了虚拟磁盘和一个可启动的操作系统的文件。在云中，镜像可以用来创建虚拟机实例。如欲了解如何创建镜像文件，请参阅*OpenStack Virtual Machine Image Guide*

如果你权限够大，你可以上传和管理虚拟机镜像。操作人员可能会把上传和管理镜像的权限限制在云管理员和操作人员的角色上。如果你有相应权限，在控制台里，你可以在管理员Project中上传和管理镜像。

> 注意：
> 你还可以用 `glance` 和 `nova` 这两个命令行工具，或者“Image service and Compute APIs”来管理镜像。

上传镜像

要把镜像上传至某个Project，请参考如下步骤：

- 登录控制台。
- 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
- 在Project选项卡种，打开Compute选项卡，然后点击Images分类。
- 点击Create Image。
  此时将会弹出一个“Create An Image”对话框。
- 输入如下值：

选项	说明
Name	镜像的名称
Description	镜像的概要介绍
Image Source	在下拉菜单中选择镜像源。有镜像地址和镜像文件可供选择。
Image File or Image Location	根据用户在“Image Source”中的选择，在此处可以填写镜像的URL地址，或者浏览本地文件寻找镜像。
Format	选择镜像的格式（如“QCOW2”格式等）
Architecture	指定该镜像的架构。例：32位的架构是“i386”，64位的架构是“x86_64”
Minimum Disk (GB) and Minimum RAM (MB)	此处请留空
Copy Data	勾选此项表示希望将镜像数据复制到镜像服务中
Public	勾选此项表示此镜像会对此Project下的所有用户公开
Protected	勾选此项表示只有有权限的人才能删除此镜像

- 点击“创建镜像”。
  该镜像将会进入队列等待上传。从进入队列到启用状态，可能需要一定的时间。

更新镜像

要更新一个已有的镜像，请参考如下步骤：

- 登录控制台。
- 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
- 在Project选项卡种，打开Compute选项卡，然后点击Images分类。
- 选择你想修改的镜像。
- 在Action列，点击More，然后在列表中点击Edit Image。
- 在Update Image对话空中，可以进行如下操作：
  - 重命名镜像。
  - 勾选Public，将该镜像公开。
  - 取消勾选Public，将该镜像转为私有。
- 点击Update Image。

删除镜像

删除镜像将永久删除，不能恢复。只有拥有相应权限的用户才能删除镜像。

- 登录控制台。
- 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
- 在Project选项卡种，打开Compute选项卡，然后点击Images分类。
- 选择想删除的镜像。
- 点击Delete Images。
- 在确认删除的对话框中，点击Delete Images来确认删除。

权限和安全配置

在创建服务器实例之前，你需要在该实例上添加安全组规则，这样服务器实例的用户才能ping这个主机或者通过SSH连接这个实例。安全组是IP过滤的规则组，它决定了网络的连通性，适用于一个Project内的所有实例。配置安全组有两种方式：要么是在默认的安全组中添加规则，要么新建一个安全组，然后把规则写在新的安全组里。

密钥对是SSH的证书，在实例启动时被注入进系统里。如欲使用密钥对注入，镜像中的系统上必须安装了cloud-init包。每个Project必须有至少一个密钥对，欲知详情，请参看添加密钥对。

如果你使用了其他工具生产了密钥对，你可以将这个密钥对导入进OpenStack。密钥对可以用在一个Project下的多个实例里。欲知详情，请参看导入密钥对。

> 注意：
> 一对密钥只属于一个独立的用户，而不属于Project。如果想让多个用户都用同一个密钥对的话，需要让每个用户都导入一次才可以。

在OpenStack中创建实例时，OpenStack会在可用的IP段中分配一个固定IP给这个实例。在这个实例的整个生命周期里，这个IP都和它相关联，只有删掉这个实例后这个IP才被释放。不过除了固定IP外，Floating IP也可以和实例相关联。和固定IP不同的是，floating IP可以无视实例的状态，随时被关联到一个不同的实例上。

在默认的安全组上添加规则

以下操作过程介绍了如何在一个实例上启用SSH和ICMP（ping）权限。该操作对Project内的所有实例生效，也建议用户在所有的Project下执行这些操作，除非你们有什么特殊理由要禁用实例的SSH和ICMP。

如果你的云需要，也可以调整以下的步骤，从而添加别的安全组规则。

> 注意：
> 添加规则的时候，必须指定源端口或目的端口使用的协议。

1. 登录控制台。
2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
3. 在Project选项卡中，打开Compute选项卡，点击 `Access & Security` 分类。在这里，安全组选项列出了当前Project适用的安全组信息。

4. 选择默认的安全组，点击“规则管理”。
5. 点击“添加规则”。
6. 在添加规则对话框，输入如下规则：

```
Rules: SSH
Remote: CIDR
```

> 注意：
> 如果要只允许某一个IP段内的主机访问，在CIDR框中填写相应的IP段即可。

7. 点击“添加”。
   此时该Project下的所有实例的SSH端口（22）便都打开了。

8. 再次点击“添加规则”。
9. 在添加规则对话框，输入如下规则：

```
Rules: All ICMP
Remote: Ingress
```

10. 点击“添加”。
    此时该Project下的实例应该可以接收ICMP包了。

添加密钥对

至少要给每一个Project都至少创建一个密钥对：

1. 登录控制台。
2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
3. 在Project选项卡中，打开Compute选项卡，点击 `Access & Security` 分类。
4. 点击密钥对选项卡，该选项卡内会显示该Project内所有可用的密钥对。
5. 点击创建密钥对。
6. 在创建密钥对窗口，输入密钥对的名字，然后点击创建密钥对。
7. 下载生成的密钥对。

导入密钥对

1. 登录控制台。

2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
3. 在Project选项卡中，打开Compute选项卡，点击 `Access & Security` 分类。
4. 点击密钥对选项卡，该选项卡内会显示该Project内所有可用的密钥对。
5. 点击导入密钥对。
6. 在导入密钥对对话框，输入你要导入的密钥对的名字，将公钥粘贴进公钥框，然后点击导入密钥对。
7. 在本地保存 `*.pem` 文件。
8. 如果要将私钥修改成只有你能访问，使用如下命令：

```
$ chmod 0600 yourPrivateKey.pem
```

> 注意：
> 如果你在用Windows访问OpenStack控制台，用PuTTYgen导入 `*.gem` 文件，然后转换成 `*.ppk` 。欲知详情，请参阅WinSCP web page for PuTTYgen。

9. 通过运行 `ssh-add` 来导入密钥。

```
$ ssh-add yourPrivateKey.pem
```

此时你电脑上的电脑便注册了该密钥对的公钥。

在OpenStack控制台中，所有密钥对都会在Access & Security选项卡中列出。

为服务器实例分配Floating IP

在OpenStack中创建实例时，OpenStack会在可用的IP段中分配一个固定IP给这个实例。在这个实例的整个生命周期里，这个IP都和它相关联，只有删掉这个实例后这个IP才被释放。

不过，除了固定IP外，Floating IP也可以和实例相关联。和固定IP不同的是，floating IP可以在无视实例状态的情况下随时被关联到一个不同的实例上。

以下步骤详述了将一个IP从IP资源池中预留出来变成Floating IP，和把这个IP分配给某个实例的过程。

1. 登录控制台。
2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。

3. 在Project选项卡中，打开Compute选项卡，点击 `Access & Security` 分类。
4. 点击Floating IP选项卡，该选项卡中会显示所有分配给实例的Floating IP。
5. 点击“Allocate IP To Project”。
6. 选择IP地址的来源IP池。
7. 点击“Allocate IP”。
8. 在“Floating IPs”列表中，点击“Associate”。
9. 在“Manage Floating IP Associations”对话框中，选择如下选项。

    - “IP Address”一栏是自动填好的，但是你也可以用“+”按钮添加一个新的IP地址。
    - 在“Port to be associated”一栏，在列表中选择一个端口。

      这个列表列出了所有实例和这些实例绑定的固定IP地址。

10. 点击“Associate”。

> 注意：
> 如果要解绑IP，请点击“Disassociate”按钮。

如果要将Floating IP重新放回IP资源池，点击“More”，然后点击“Release Floating IP”。

创建和管理实例

服务器实例是运行在云中的虚拟机。用户可以通过下面几个资源创建一个新的实例。

- 上传到OpenStack的镜像。
- 复制到persistent volume中的镜像。该volume须是由 `cinder-volumn` 通过iSCSI提供的。实例可以在该volume上启动。

创建实例

如果要从volume中创建实例，请参考如下步骤：

1. 在选择在哪一个volume上启动时，先在这个volume中选择任意一个image然后选择启动启动。但其实并不是在你选择的镜像上启动的。具体的启动镜像是在你下一步选择镜像时才决定的。

   如果要从volume中启动一个Xen镜像，你在

这一章节太奇葩。我先把别的翻译完，然后实际操作一下，然后再来翻译这一章。

创建和管理网络

OpenStack网络服务是一个在OpenStack云内即可使用的、可扩展的管理网络的工具。这个工具能够帮助用户快速响应网络变更的需求（例如创建和分配新的IP地址等）。

OpenStack中的网络很复杂。本章仅简单介绍了如何创建网络和路由。欲了解详细的OpenStack网络管理内容，请参阅OpenStack Cloud Administrator Guide。

创建网络

1. 登录控制台。
2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
3. 在Project选项卡中，打开Network选项卡，点击Network分类。
4. 点击“创建网络”。
5. 在“创建网络”对话框，填入以下内容
    - Network选项卡
        - Network Name：为新建的网络取个名字
        - Admin State：选择要在什么state下启动该网络
    - Subnet选项卡
        - Create Subnet：勾选这个框表示来创建一个子网
          创建网络的时候不需要必须指定子网，但是如果不指定子网的话，任何与之关联的服务器实例都会收到一个报错信息。
        - Subnet Name：为子网指定一个名称。
        - Network Address：指定子网的IP地址。
        - IP Version：选择IPv4或者IPv6。
        - Gateway IP：指定一个IP为网关地址。该配置可省略。
        - Disable Gateway：禁用网关。
    - Subnet Details选项卡
        - Enable DHCP：勾选此处则启用DHCP。
        - Allocation Pools：指定IP地址池。
        - DNS Name Servers：指定DNS的地址。
        - Host Routes：指定主机路由的IP地址。
6. 点击“创建”

```
此时OpenStack便在Network选项卡上显示网络信息了。
```

创建路由器

1. 登录控制台。
2. 在Project选项卡中，进入CURRENT PROJECT，然后选择目标Project。
3. 在Project选项卡中，打开Network选项卡，点击Network分类。
4. 点击“Create Router”。
5. 在“Create Router”对话框，给新路由器起一个名字，然后点击“Create Router”。
   现在这个新的路由器已经在Router选项卡中列出了。
6. 点击新路由的“Set Gateway”选项。
7. 在“External Network”框，指定该路由器要连接到哪个网络上，然后点击“Set Gateway”。
8. 如果要将一个私网连接在刚刚创建的路由器上，请遵循以下步骤：
   i. 在“Router”选项卡上，点击路由器的名字。
   ii. 在“Router Details”页面，点击“Interface”选项卡，然后点击“Add Interface”。
   iii. 在“Add Interface”对话框，选择一个Subnet。
   此外，在“Add Interface”对话框，你可以为连接在这个子网的路由器接口设置一个IP Address。
   如果你执意不设置IP Address，OpenStack Networking会把这个子网中的第一个IP地址分配给这个路由器接口。 Router Name和Router ID这两栏会自动更新。
9. 点击“Add Interface”。

至此，我们创建了一个新的路由器。在Network Torpology选项卡中可以查看新的网络拓扑图。

创建和管理对象容器

OpenStack对象存储是一个分布式的对象存储平台，支持API调用。该功能可以直接结合进其他应用中，也可以用来存储各种文件，包括虚拟机镜像、备份、归档、甚至是媒体文件。在OpenStack控制台，用户暂时只能管理容器和镜像。

在OpenStack对象存储中，容器是存放对象文件的空间，就像Windows或者Linux的文件夹是存放文件的空间一样。只是容器和容器不能像文件夹一样嵌套。OpenStack的对象包括存储的文件和各种附属的元数据。

创建容器

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 点击Create Container。
5. 在Create Container对话框，为新的容器起个名字，然后点击Create Container。

至此，您已成功创建了一个新容器。

> 注意： 如果要删除容器，点击More，然后选择Delete Container。

上传对象

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 选择一个容器来存放对象。
5. 点击Upload Object。
6. 此时会弹出一个对话框，名为“Upload Object To Container: ”。其中处是存放本次上传对象的容器名。
7. 输入对象的名称。
8. 找到你想上传的本地文件。
9. 点击Upload Object。

至此，您成功地在容器中上传了一个对象。

管理对象

编辑对象

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 选择一个容器来存放对象。
5. 点击More，在下拉菜单中点击Edit。
6. 此时会弹出一个对话框，名为“Edit Object”。
7. 找到你想上传的本地文件。（译注：上传的新对象会覆盖掉原来的旧对象。）
8. 点击Update Object。

将某对象在不同容器之间复制

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 找到想要复制的文件对象。
5. 点击右边的“More”，然后在下拉菜单中点击“复制”。
6. 在Copy Object对话框，输入如下项：
    - 目标容器：选择复制的目的地。
    - 路径：为复制过去的对象指定在新容器中的路径。
    - 目标对象名：为复制过去的对象设置新的对象名称。
7. 点击Copy Object。

创建一个只有元数据，但没有文件的对象

在容器中，可以创建一个只有元数据，但没有文件的对象。待文件可用时再将文件上传至对象即可。这样的临时对象起到了占位的作用，还便于提前将的对象的元数据和URL共享出去。

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 选择一个容器来存放对象。
5. 点击Upload Object。
6. 此时会弹出一个对话框，名为“Upload Object To Container: ”。其中处是存放本次上传对象的容器名。

7. 输入对象的名称。
8. 点击Upload Object。

创建一个Pseudo-folder

Pseudo-folders是和正常操作系统类似的文件夹。只是他们是由对象的前缀（或前缀集合）标示的。

1. 登录OpenStack控制台。
2. 在Project选项卡中的CURRENT PROJECT中，选择相应的Project。
3. 在Project选项卡中打开Object Store选项卡，然后点击Containers分类。
4. 选择希望操作的Container。
5. 点击Create 为Pseudo-folder起个名字。 此时会弹出一个对话框，名为“Create Pseudo-Folder in Container ”，其中是你希望操作的容器名称。
6. 为Pseudo-folder起个名字。 在对象名称中，OpenStack会用一个斜线“/”来划分Pseudo-folder和真实的对象名称。
7. 点击Create

创建和管理卷

卷是一种块存储设备。我们把卷挂载在实例实例上，从而达到持久化存储的目的。您可以随时将某个卷安装在一个运行中的实例上，或者从实例上卸载卷，再把它安装再另一个实例上。您还可以给卷拍快照，以及删除卷等等。但只有管理员才能指定卷的类别。

创建卷

1. 登录控制台。
2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 点击Create Volume。在打开的对话框中填入如下信息。
    - Volume Name：为卷指定名称。
    - Description：非必须填写，给该卷添加一段简短的描述。
    - Type：此处请留空。
    - Size(GB)：该卷的大小（以GiB为单位）。
    - Volume Source：在如下选项中选择。
        - No source, empty volume：创建一个空卷，其中不包含文件系统和分区表。
        - Image：如果选择此项，一个叫“Use image as a source”的下拉框便会显示出来，您可以在此选择镜像。
        - Volume：如果选择此项，一个名为“Use volume as a sourc”的下拉框便会显示出来。您可以在此选择卷。快照和卷的选项只会在快照和卷真实存在的情况下才会显示出来。
    - 可用区域：在列表中选择可用区域。默认的情况下，可用区域的设定是由云供应商决定的（例如，`us-west` 或者 `apac-south` ）。有些时候，这个地方也可能是 `nova` 。
5. 点击Create Volume。

将卷安装在实例上

在你创建了卷之后，此时便可以把卷安装在实例上了。不过，一次只能安装一个卷。

1. 登录控制台。

2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 选择想要安装在实例上的卷，然后点击“Edit Attachments”。
5. 在“Manage Volume Attachments”对话框，选择要安装的实例。
6. 输入一个设备名称，以便实例通过这个名称访问到卷。

> 注意： 有时由于虚拟机设置原因，有些设备的实际名称和卷名称是不一样的。

1. 点击Attach Volume。

   此时控制台会上便会列出实例信息，以及实例连接的卷信息。

现在您便可以在控制台下卷选项卡中上看见卷的状态了，该卷有两种状态：要么是Available，要么是In-Use。

现在您便可以在实例中使用这个磁盘了。挂载，格式化，您随意。

Detach a volume from an instance

1. 登录控制台。
2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 选择卷，点击“Edit Attachments”。
5. 点击“Detach Volume”并确定。

此后会有一条消息提示您该变更是否成功完成。

为卷创建快照

1. 登录控制台。
2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 选择想要创建快照的卷。
5. 点击“More”，然后选择“Create Snapshot”。
6. 在弹出的对话框中输入快照名称和简介。
7. 确定变更。

此时，在控制台下的“Volume Snapshot”中，新的卷快照便会显示出来。

编辑卷

1. 登录控制台。
2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 选择想要编辑的卷。
5. 在Action列中，点击“Edit Volume”。
6. 在“Edit Volume”对话框中，编辑该卷的名字和描述。
7. 点击“Edit Volume”。

> 注意： 想要给某个卷扩容，请点击“More”菜单，然后选择“Extend Volume”选项，然后再弹出的框中输入新的卷大小。

删除卷

如果您删除了一个实例，安装在这个实例上的卷中的文件是不会被删除的。

1. 登录控制台。
2. 在PROJECT选框中选择希望操作的Project。
3. 在PROJECT选项卡种，打开Compute选项卡，然后点击Volumes子项。
4. 选择想要删除的卷。此处可以多选。
5. 点击删除卷，然后确认操作。

此后会有一条消息提示您该变更是否完成。

创建和管理共享文件夹（？）

概览

每一个OpenStack项目都会提供命令行工具，这样一来，用户通过简单的命令便可使用项目的API了。比如，Compute服务就提供了nova这个命令行工具。

您可以在命令行里直接输入命令，也可以把这些命令写进脚本，或者一些自动化任务中。如果您有必要的身份认证信息（例如账号和密码），你还可以在远程使用这些命令。

这些命令行工具实际上是cURL+API请求的集合。OpenStack的API遵循RESTful API规范，运行在HTTP协议下。这些API包含了HTTP方法（method），URI，媒体类型（media type）和返回码（response code）。

OpenStack的API都是开源的Python客户端，能运行在Linux和Mac OS X上。有些API请求允许您添加debug参数，添加之后这些API请求便会显式地打印出来。这样会让您更加了解OpenStack的API调用。

作为终端用户，您可以使用OpenStack控制台查看分配给您的各种资源。此外，您可以使用本章的例子来创建其他种类和大小的服务器实例。

OpenStack服务和客户端。

服务	客户端	包	简介
Application catalog	murano	python-muranoclient	创建和管理应用
Block Storage	cinder	python-cinderclient	创建和管理卷
Compute	nova	python-novaclient	创建和管理镜像，实例以及系统型号
Containers service	magnum	python-magnumclient	创建和管理容器
Database service	trove	python-troveclient	创建和管理数据库
Data processing	sahara	python-saharaclient	在OpenStack上创建和管理Hadoop集群
Deployment service	tuskar	python-tuskarclient	安排部署
		python-	创建和管理用户，租户，角

Identity	keystone	keystoneclient	色，endpoint和认证信息
Image service	glance	python-glanceclient	创建和管理镜像
Key Manager service	barbican	python-barbicanclient	创建和管理密钥
Monitoring	monasca	python-monascaclient	监控方案
Networking	neutron	python-neutronclient	管理实例的网络配置
Object Storage	swift	python-swiftclient	对象存储服务中的统计信息收集，内容展示，元数据更新，文件的上传下载删除。（Gain access to an Object Storage installation for ad hoc processing
Orchestration	heat	python-heatclient	从templates中启动stack，查看运行中的stack（包括事件和资源），以及更新和删除stack
Rating service	cloudkitty	python-cloudkittyclient	评价服务
Shared file systems	manila	python-manilaclient	创建和管理共享的文件系统
Telemetry	celiometer	python-celiometerclient	创建和收集OpenStack中的各种指标
Telemetry v3	gnocchi	python-gnocchiclient	创建和收集OpenStack中的各种指标
Workflow service	mistral	python-mistralclient	OpenStack云服务的工作流系统
Common client	openstack	python-openstackclient	OpenStack项目的普适的客户端

安装OpenStack命令行工具

让我们先来了解下如何安装OpenStack依赖的软件和Python包。

安装依赖的软件

绝大多数Linux发行版都为您提供了可以直接安装的命令行工具包，详情请参见Installing from packages。

如果您需要用源码包安装这些命令行工具。下面的列表列出了您需要提前安装的软件，以及一些必要的安装提示。

- Python 2.7及其后续版本（但并不支持Python 3）
- setuptools
    - Mac OS X：已经默认安装好了。
    - Linux：绝大多数Linux发行版都提供了setuptools的安装包。您在您系统默认的软件包管理器上应该都能找到。如果找不到，您还可以直接在以下链接处下载到：https://pypi.python.org/pypi/setuptools
    - Windows：如果您要在Windows下安装setuptools，我们建议您参阅setuptools的网站：https://pypi.python.org/pypi/setuptools/。此外，您也可以下载由Christoph Gohlke维护的非官方安装包：http://www.lfd.uci.edu/~gohlke/pythonlibs/#setuptools。
- pip package

    - 在Linux，Mac OS X以及Windows上安装命令行工具，您都需要用到pip。pip使用起来很简单，并且能确保您下载到的一定是最新版的客户端，还能让您在日后方便地更新或删除您下载的包。

      因为安装的过程中需要编译源码，因此在安装中您的系统上要有适用的Python开发包。

      通过包管理器来安装pip：

      Mac OS

      ```
      # easy_install pip
      ```

      Microsoft Windows

使用之前请确保 `C:\Python27\Scripts\这个文件夹路径在环境变量 PATH 中。然后通过` easy_install```命令来安装pip。 此外，您也可以下载和使用由Christoph Gohlke维护的非官方安装包：
http://www.lfd.uci.edu/~gohlke/pythonlibs/#setuptools。

Ubuntu和Debian

```
# apt-get install python-dev python-pip
```

如果您用的操作系统版本不同，您可能需要不同的包。具体需要哪些包和您要安装哪些包有关，比如Tempest。

Red Hat Enterprise Linux, CentOS 或者 Fedora

您可以通过如下命令安装：

```
# yum install python-devel python-pip
```

RDO也提供了客户端的安装包，您可以参看Installing from packages，其中介绍了如何让yum获取RDO中的安装包。

SUSE Linux Enterprise Linux 11

在Open Build Service上有一个封装好的pip包，可以让您通过zypper安装pip。首先，您需要添加Open Build Service：

```
# zypper addrepo -f obs://Cloud:OpenStack: \
kilo/SLE_12 Kilo
```

然后您就可以用zypper安装pip了：

```
# zypper install python-devel python-pip
```

如同Installing from packages描述的一样，这些OpenStack客户端也有直接封装好的版本，可以通过zypper直接安装。

openSUSE

您可以通过zypper直接安装pip：

```
# zypper install python-devel python-pip
```

如同Installing from packages描述的一样，这些OpenStack客户端也有直接封装好的版本，可以通过zypper直接安装。

安装命令行客户端

在您参照本小节操作时，请将命令中的 `"PROJECT"` 替换为要安装的客户端的名字，例如 `“nova”` 。您安装每一个客户端，都要用这样的做法。以下值是有效的：

- barbican - 密钥管理服务API
- ceilometer - 遥测（？）API
- cinder - 块存储API及扩展。
- cloudkitty - 评价服务API
- glance - 镜像服务API
- gnocchi - 遥测服务APIv3
- heat - 统一编排API
- magnum - 容器服务API
- manila - 共享文件系统服务API
- mistral - 工作流服务API
- monasca - 监控API
- murano - 应用系列API
- neutron - 网络API
- nova - 计算API和扩展
- sahara - 数据处理API
- swift - 对象存储API
- trove - 数据库服务API
- tuskar - 部署服务API
- openstack - 支持多个OpenStack服务的全能型的命令行工具

以下命令行客户端已逐渐被那个叫openstack的全能型命令行替代。

- keystone - 身份认证服务API及扩展

如下例子介绍了如何通过命令行，用pip安装nova客户端。

```
# pip install python-novaclient

##### 通过pip安装

我们强烈建议您用pip来安装OpenStack命令行客户端，不管您用的是Linux, Mac OS X

每一个客户端都用下面这条命令来安装：

- Mac OS X或者Linux：
```

pip install python-PROJECTclient

```

- Microsoft Windows：
```

C:>pip install python-PROJECTclient

```

##### 用包管理器安装

RDO, openSUSE, SUSE Linux Enterprise, Debian和Ubuntu版本可以在不安装pip

- 在RHEL, CentOS, Fedora下，用yum来安装托管在RDO上的安装包：
```

yum install python-PROJECTclient

```

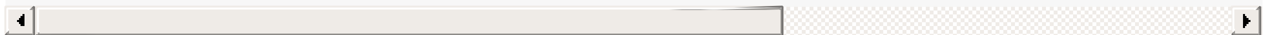
- 在Debian或Ubuntu系统，用apt-get来安装各个包：
```

apt-get install python-PROJECTclient

```
-在openSUSE下，您可以使用zypper从软件包服务中下载到各个客户端包：
```

zypper install python-PROJECTclient

```
- 如果您用的时SUSE Linux Enterprise Server，您也可使用zypper来安装这些包。
```



zypper addrepo -f obs://Cloud:OpenStack:Kilo/SLE_12 Kilo

```
然后您就可以通过如下命令安装各个包了：
```

zypper install python-PROJECTclient

```
#### 升级或卸载命令行客户端

如果您要升级某个包，您可以在pip命令上添加参数：
```

pip install --upgrade python-PROJECTclient

```
如果您要卸载某个包，请使用```pip uninstall```命令：
```

pip uninstall python-PROJECTclient

```

## 下一步要做的事

在您运行命令之前，您必须创建一个叫PROJECT-openrc.sh的文件，然后运行一次，以设置环境变量。请参见Set environment variable using the OpenStack RC file。
```

查看某个命令行客户端的版本号

运行以下命令来查看某个命令行客户端的版本号：

```
$ PROJECT --version
```

例如，如果想要查看nova的版本号，执行以下指令：

```
$nova --verion
2.31.0
```

用OpenStack RC文件设置环境变量

为了给OpenStack命令行客户端提供所需的环境变量，您必须创建OpenStack RC文件，或者说叫 `openrc.sh` 文件。如果您安装的时候有这个文件，您可以在OpenStack控制台用任意一个用户下载到这个文件。这个针对某个project有效地文件包含了所有OpenStack服务所需的关键信息。

当您手动载入这个文件时，里面的环境变量配置便会加载进当前Shell中。这些环境变量连接了本地的OpenStack客户端和在云上的OpenStack服务。

> 注意： 在Microsoft Windows中，我们并不会用环境变量文件导入环境变量。环境变量一般都是在*System Properties*对话框中的*Advaned*选项卡中设置的。

下载和载入OpenStack RC文件

登录OpenStack控制台，选择对应的Project，然后依次点击“Compute”和“Access&Security”。

在API Access选项卡中，点击Download OpenStack RC文件，把文件下载下来。这个文件名的格式是“ `PROJECT-openrc.sh` ”，此处的PROJECT便是你下载时所在的那个Project。

将这份PROJECT-openrc.sh文件复制到您想运行OpenStack命令的机器上。（比如您想用glance工具从某机器上上传一个图片到云，那就把这个文件复制到这机器上。）

然后载入这个文件，请看下面这个例子，假定我们要设置demo这个Project。

```
$ source demo-openrc.sh
```

当系统要求您输入密码时，您要输入刚刚下载这个文件时登录进OpenStack那个用户的密码。

手动创建和载入OpenStack RC文件

此外，如果您无法在OpenStack控制台处下载到PROJECT-openrc.sh，您可以按照下面的教程手动创建一个。

在文本编辑器中，创建一个叫PROJECT-openrc.sh的文件，添加如下认证信息。

```
export OS_USERNAME=username
export OS_PASSWORD=password
export OS_TENANT_NAME=projectName
export OS_AUTH_URL=https://identityHost:portNumber/v2.0
# 以下内容也可以省略
export OS_TENANT_ID=tenantIDString
export OS_REGION_NAME=regionName
export OS_CACERT=/path/to/cacertFile
```

在你想运行OpenStack命令的shell中加载这个文件。本例中，我们为admin这个project加载了admin-openrc.sh。

```
$ source admin-openrc.sh
```

> 注意：
> 如果您操作的时候使用了本例中的RC文件，在运行OpenStack命令的时候系统就不会再要求您输入密码了，因为密码已经用明文存储在这个RC文件中了。请给此文件设置一个合适的权限来避免安全问题。您也可以在删除该RC文件中有关密码的条目，在运行OpenStack时使用“`--password`”参数，这样系统就会在执行命令时向您询问密码了。
>
> 注意：
> 如果在`OS_AUTH_URL`中设置了使用HTTPS协议的话，您必须设置OS_CACERT这个环境变量，因为TLS（HTTPS）服务器认证过程中需要用到这个环境变量。该环境变量所指向的证书在我们认证服务器证书时会用到。

让环境变量值失效

如果您使用OpenStack命令，您可以用命令中的参数覆盖掉某些环境变量值，这些可选参数可以用各个客户端的 `help` 命令查询到。例如，您可以在使用 `openstack` 命令时，用添加参数的方法让环境变量 `PROJECT-openrc.sh` 中的 `OS_PASSWORD` 失效：

```
$ openstack --os-password PASSWORD service list
```

本例中 `PASSWORD` 是您的密码。

您在用任何命令行工具与OpenStack交互时都需要提供用户名和密码。这些用户名和密码信息可以通过不同方法来指定，比如，用环境变量文件或者是命令参数。不过这两种方法都不安全。

举例：如果您在用命令行客户端时用 `--os-password` 写了密码。那任何有权限登录您电脑的人都可以用 `ps` 命令看见密码。

为了避免密码被别人看见，请考虑使用交互式的方式提供密码。

管理镜像

用户的角色是由云管理员分配的。不同的角色决定了一个用户是否有上传和管理镜像的权限。云管理员一般都会将上传和管理镜像的权限限制在管理员范围内。

您可以用 `glance` 工具上传镜像，可以用 `nova` 工具管理镜像。后者可以让您列出镜像和删除镜像，设置和删除镜像元数据，以及用快照或备份的形式创建运行中的实例的镜像。

一旦您上传了镜像，就没办法更改了。

欲了解创建镜像的内容，请参阅Virtual Machine Image Guide章节。

列出镜像，查看某个镜像的详情

要列出镜像，以及查看某个镜像的详情，请使用 `glance image-list` 和 `glance image-show` 命令。

```
$ glance image-list
+----------+---------------------------------+------------+-------
| ID       | Name                            | Disk Format | Contai
+----------+---------------------------------+------------+-------
| 397e7... | cirros-0.3.2-x86_64-uec         | ami         | ami
| df430... | cirros-0.3.2-x86_64-uec-kernel  | aki         | aki
| 3cf85... | cirros-0.3.2-x86_64-uec-ramdisk | ari         | ari
| 7e514... | myCirrosImage                   | ami         | ami
+----------+---------------------------------+------------+-------
```

```
$ glance image-show myCirrosImage

+---------------------------------------+------------------------
| Property                              | Value
+---------------------------------------+------------------------
| Property 'base_image_ref'             | 397e713c-b95b-4186-ad46-6
| Property 'image_location'             | snapshot
| Property 'image_state'                | available
| Property 'image_type'                 | snapshot
```

```
| Property 'instance_type_ephemeral_gb' | 0
| Property 'instance_type_flavorid'     | 2
| Property 'instance_type_id'           | 5
| Property 'instance_type_memory_mb'    | 2048
| Property 'instance_type_name'         | m1.small
| Property 'instance_type_root_gb'      | 20
| Property 'instance_type_rxtx_factor'  | 1
| Property 'instance_type_swap'         | 0
| Property 'instance_type_vcpu_weight'  | None
| Property 'instance_type_vcpus'        | 1
| Property 'instance_uuid'              | 84c6e57d-a6b1-44b6-81eb-f
| Property 'kernel_id'                  | df430cc2-3406-4061-b635-a
| Property 'owner_id'                   | 66265572db174a7aa66eba661
| Property 'ramdisk_id'                 | 3cf852bd-2332-48f4-9ae4-7
| Property 'user_id'                    | 376744b5910b4b4da7d8e6cb4
| checksum                              | 8e4838effa1969ad591655d64
| container_format                      | ami
| created_at                            | 2013-07-22T19:45:58
| deleted                               | False
| disk_format                           | ami
| id                                    | 7e5142af-1253-4634-bcc6-8
| is_public                             | False
| min_disk                              | 0
| min_ram                               | 0
| name                                  | myCirrosImage
| owner                                 | 66265572db174a7aa66eba661
| protected                             | False
| size                                  | 14221312
| status                                | active
| updated_at                            | 2013-07-22T19:46:42
+---------------------------------------+--------------------------
```

在查看镜像列表时，您还可以用 `grep` 用关键词过滤列表：

```
$ glance image-list | grep 'cirros'
| 397e713c-b95b-4186-ad46-612... | cirros-0.3.2-x86_64-uec
| df430cc2-3406-4061-b635-a51... | cirros-0.3.2-x86_64-uec-kernel
| 3cf852bd-2332-48f4-9ae4-7d9... | cirros-0.3.2-x86_64-uec-ramdisk
```

> 注意： 为了存储镜像的存储位置元数据，使得客户端能直接存取文件，请在 `/etc/glance/glance-api.conf` 文件中维护以下内容。
>
> - `show_multiple_localtions = True`
> - `filesystem_store_metadata_file = FILEPATH` ，其中FILEPATH指向了一个JSON文件，该文件保存了您OpenStack镜像的挂载路径和一个唯一ID：
>
> ```
> [{
>  "id": "2d9bb53f-70ea-4066-a68b-67960eaae673",
>  "mountpoint": "/var/lib/glance/images/"
> }]
> ```
>
> 您重启了镜像服务之后，您便可以通过如下的命令来查看镜像的位置信息了：
>
> ```
> $ glance --os-image-api-version 2 image-show imageID
> ```
>
> 比如，使用刚刚我们给出的image ID，您可以使用如下命令：
>
> ```
> $ glance --os-image-api-version 2 image-show 2d9bb53f-70ea-
> ```

创建或上传镜像（glance）

如果要创建镜像，请使用 `glance image-create` 命令：

```
$ glance image-create IMAGENAME
```

如果要通过名称或ID更新一个镜像，请使用 `glance image-update` ：

```
$ glance image-update IMAGENAME
```

以下列表列出了您可以在 `create` 和 `update` 时添加的参数。这些参数用于修改各种镜像属性。想了解更多详情，请参阅OpenStack Command-Line Interface Reference一文中镜像服务的章节。

- `--name NAME` 镜像的名字。
- `--disk-format DISK_FORMAT` 镜像的格式。可接受的格式有ami, ari, aki, vhd, vmdk, raw, qcow2, vdi和iso。
- `--container-format CONTAINER_FORMAT` 镜像的容器格式。可接受的格式有ami, ari, aki, bare和ovf。
- `--owner TENANT_ID --size SIZE` 哪一个用户拥有这个镜像。以及镜像的大小，单位是byte。
- `--min-disk DISK_GB` 启动这个镜像最少需要的磁盘空间，单位是Gigabyte。
- `--min-ram DISK_RAM` 启动这个镜像最少需要的没存大小，单位是Megabyte。
- `--location IMAGE_URL` 这个镜像的数据的存放位置。例如，如果这个镜像存放在swift中，那您便可以将其指定为 `swift://account:key@example.com/container/obj`
- `--file FILE` 更新镜像时需要上传的镜像文件。除此之外，您还可以通过stdin将镜像传递给客户端。
- `--checksum CHECKSUM` 镜像数据的哈希值，做校验用。
- `--copy-from IMAGE_URL` 和 `--location` 的使用方法类似，但是该项会让服务器立即将镜像数据复制到指定的镜像存储中。
- `--is-public [True|False]` 设置是否允许这个镜像对所有用户开放（默认只能给管理员用）。
- `--is-protected [True|False]` 设置该镜像能否被删掉。
- `--property KEY=VALUE` 和镜像相关联的其他设置。该参数可以多次使用。
- `--purge-props` 删除所有没在update操作中定义的选项。否则，那些没指定的都会被保留下来。
- `--human-readable` 用方便阅读的单位显示镜像大小。

下面的例子向您展示了如何在qcow2格式下上传一个CentOS 6.3镜像，且设置成公共可访问权限。

```
$ glance image-create --name centos63-image --disk-format qcow2 \
--container-format bare --is-public True --file ./centos63.qcow2
```

下面的例子向您展示了如何更新一个镜像的属性，包括硬盘种类，CD-ROM种类，和VIF模式。

```
$ glance image-update \
    --property hw_disk_bus=scsi \
    --property hw_cdrom_bus=ide \
    --property hw_vif_model=e1000 \
    f16-x86_64-openstack-sda
```

目前，libvirt虚拟化工具是根据虚拟机管理器的种类来决定硬盘，CD-ROM和VIF设备的模式的， `libvirt_type` 设置在 `/etc/nova/nova.conf` 中。为了优化性能，libvirt默认会把disk和VIF（NIC）都配置成virtio。这样做的缺点是，对于那些没有virtio驱动的操作系统来说，就根本跑不起来系统了，比如BSD，Solaris和一些老版本的Linux和Windows。

如果您指定了一个不支持的硬盘或光盘格式，请参阅Disk and CD-ROM bus model values table。如果您指定了一个不支持的VIF格式，导致实例不能启动，请参阅VIF model values table。

可用的模式取决于libvirt_type设置，如下表所示。

磁盘和光盘总线设置

libvirt_type setting	Supported model values
qemu or kvm	ide/scsi/virtio
xen	ide/xen

VIF模式设置

libvirt_type setting	Supported model values
qemu or kvm	e1000/ne2k_pci/pcnet/rtl8139/virtio
xen	e1000/netfront/ne2k_pci/pcnet/rtl8139
vmware	VirtualE1000/VirtualPCNet32/VirtualVmxnet

解决创建镜像过程中出现的问题

如果您在使用Image或Compute创建镜像时遇到了问题，以下信息也许会帮您解决问题：

- 首先，保证您使用的 `qemu` 版本在0.14版以上。这之前的版本会在 `nova-compute.log` 文件中留下 `unknown option -s` 的报错。
- 在 `/var/log/nova-api.log` 和 `/var/log/nova-compute.log` 文件中查看错误信息。

管理卷

卷是可装可卸的块存储设备，有点类似U盘。您一次只能把一个卷装在一个实例上。要创建和管理卷，您可以通过 `nova` 和 `cinder` 客户端命令。

迁移卷

如果您是系统管理员，您可以给卷和里面连带的数据做迁移，而且这整个过程都是用户无感知、业务无感知的。不过，您只能迁移没有被装载的卷，而且卷上不能有快照。

数据迁移的可能原因有如下几种：

- 在不影响业务的情况下，关掉物理存储设备进行维护。
- 修改卷的属性。
- 腾空间。

迁移卷的时候，请使用 `cinder migrate` 命令。如下例所示：

```
$ cinder migrate volumeID destinationHost --force-host-copy True|Fa
```

在本例中， `--force-host-copy True` 会强制使用基于主机的迁移机制，而不使用驱动的优化。

> 注意： 如果该卷正在被使用，或者该卷上有快照，那目标主机是不会接受这个迁移的。如果执行迁移的用户不是管理员，迁移也会失败。

创建卷

本节我们用例子来展示如何用镜像来创建一个名为“my-new-volume”的卷。

1. 列出镜像，记下您创建卷想用的镜像的ID。

```
$ nova image-list

+----------------------+----------------------------------+--------
| ID                   | Name                             | Status
+----------------------+----------------------------------+--------
| 397e713c-b95b-4186... | cirros-0.3.2-x86_64-uec          | ACTIVE
| df430cc2-3406-4061... | cirros-0.3.2-x86_64-uec-kernel   | ACTIVE
| 3cf852bd-2332-48f4... | cirros-0.3.2-x86_64-uec-ramdisk  | ACTIVE
| 7e5142af-1253-4634... | myCirrosImage                    | ACTIVE
| 89bcd424-9d15-4723... | mysnapshot                       | ACTIVE
+----------------------+----------------------------------+--------
```

1. 列出可用的域（zone），记下您创建卷时想用的可用域的ID。

```
$ cinder availability-zone-list

+------+-----------+
| Name |   Status  |
+------+-----------+
| nova | available |
+------+-----------+
```

1. 创建一个8 GiB的空间，同时指定这个卷用到的可用域和镜像。

```
$ cinder create 8 --display-name my-new-volume --image-id 397e713c-

+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2013-07-25T17:02:12.472269      |
| display_description |                 None                 |
|     display_name    |            my-new-volume             |
|          id         | 573e024d-5235-49ce-8332-be1576d323f8 |
|       image_id      | 397e713c-b95b-4186-ad46-6126863ea0a9 |
|       metadata      |                  {}                  |
|         size        |                  8                   |
|     snapshot_id     |                 None                 |
|     source_volid    |                 None                 |
|        status       |               creating               |
|     volume_type     |                 None                 |
+---------------------+--------------------------------------+
```

1. 想要查看您的卷是否创建成功，请列出可用卷：

```
$ cinder list

+-----------------+-----------+-----------------+------+----------
|    ID           |   Status  |   Display Name  | Size | Volume Typ
+-----------------+-----------+-----------------+------+----------
| 573e024d-523... | available |  my-new-volume  |  8   |     None
| bd7cf584-45d... | available | my-bootable-vol |  8   |     None
+-----------------+-----------+-----------------+------+----------
```

如果您的卷创建成功，那这个卷的状态应该是 `available` 。如果状态是 `error` ，您很有可能给卷的大小分得太多，超过您的配额了。

将卷装载在实例上

1. 要将您的卷装载在服务器上，您需要指定服务器ID和卷ID：

```
$ nova volume-attach 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 573e024d-

+----------+--------------------------------------+
| Property | Value                                |
+----------+--------------------------------------+
| device   | /dev/vdb                             |
| serverId | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| id       | 573e024d-5235-49ce-8332-be1576d323f8 |
| volumeId | 573e024d-5235-49ce-8332-be1576d323f8 |
+----------+--------------------------------------+
```

请记录下您卷的ID。

1. 查看您卷的信息。

```
$ cinder show 573e024d-5235-49ce-8332-be1576d323f8
```

以下信息便是返回结果，其中显示了这个卷被装载到了ID为 `84c6e57d-a6b1-44b6-81eb-fcb36afd31b5` 的机器上，所在的可用区是nova，而且是可引导启动的。

重设卷的大小

1. 如果要重设某个卷的大小，前提是该卷没有被装载在任何实例上。使用卸载的命令时您需要提供实例ID和卷ID，命令如下：

```
$ nova volume-detach 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5   573e024
```

1. 列出实例：

```
$ cinder list
+-----------------+-----------+-----------------+------+------------
|       ID        |   Status  |   Display Name  | Size | Volume Type
+-----------------+-----------+-----------------+------+------------
| 573e024d-52... | available |  my-new-volume  |  8   |     None
| bd7cf584-45... | available | my-bootable-vol |  8   |     None
+-----------------+-----------+-----------------+------+------------
```

此时该卷已变为可用状态。

1. 使用如下命令来重新设定卷的大小。您需要将卷ID和新的大小作为参数传递给扩容命令。而且，调整后的大小必须要比调整前的大：

```
$ cinder extend 573e024d-5235-49ce-8332-be1576d323f8 10
```

删除卷

1. 要删除卷，您同样要保证该卷没有被装载在任何实例上。如果在服务器上卸载卷、查看卷列表，请参阅Resize a volume章节的第一步和第二步。

用卷名称或ID来指定你要删除哪个卷：

```
$ cinder delete my-new-volume
```

1. 再次列出卷列表，您会发现您的卷的状态已经变为 `deleting` 。

```
$ cinder list
+-----------------+-----------+-----------------+------+------------
|       ID        |   Status  |   Display Name  | Size | Volume Typ
+-----------------+-----------+-----------------+------+------------
| 573e024d-523... |  deleting |  my-new-volume  |  8   |     None
| bd7cf584-45d... | available | my-bootable-vol |  8   |     None
+-----------------+-----------+-----------------+------+------------
```

1. 当该卷被完全删除后，它就不会显示在卷列表中了。

```
$ cinder list
+-----------------+-----------+-----------------+------+----------
|       ID        |   Status  |   Display Name  | Size | Volume Typ
+-----------------+-----------+-----------------+------+----------
| 573e024d-523... |  deleting |  my-new-volume  |  8   |     None
| bd7cf584-45d... | available | my-bootable-vol |  8   |     None
+-----------------+-----------+-----------------+------+----------
```

转移卷

您可以用 `cinder transfer*` 命令将某个卷从一个所有者转移到另一个所有者。卷的出让人，或者说原来的主人，创建一个转移请求，然后把卷ID和认证码发给卷的受让人。受让人，或者说新主人，用卷的ID和认证码接受转移。

> 注意：
> 转移卷的操作一般只能在同一个云的内部完成，出让人和受让人必须在同一个云里。

转移卷通常发生在以下场景：

- 您创建了一个可引导卷，或者一个装了很多内容的卷，现在要把它转移给顾客。
- 向云中批量上传数据时；数据导入系统创建了一个新的块存储设备时；将数据从物理存储转移到云中时；或者将设备所有权转移给最终用户时，等等。

创建卷转移请求

1. 以卷的出让人身份登录，列出所有可用卷：

```
$ cinder list
+-----------------+-----------+--------------+------+-------------
|       ID        |   Status  | Display Name | Size | Volume Type
+-----------------+-----------+--------------+------+-------------
| 72bfce9f-cac... |   error   |     None     |  1   |     None
| a1cdace0-08e... | available |     None     |  1   |     None
+-----------------+-----------+--------------+------+-------------
```

1. 针对某一个卷，创建一个卷转移授权码

```
$ cinder transfer-create volumeID
```

该卷必须处在 `available` 状态，否则该请求会被驳回。如果转移请求在数据库中有效（也就是说它既没有超时也没有被删掉），则该卷会被标记为 `awating transfer` 状态，例如：

```
$ cinder transfer-create a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f
```

返回的结果如下所示，其中id项是本次的转移ID，授权码也显示出来了：

```
+------------+--------------------------------------+
|  Property  |                Value                 |
+------------+--------------------------------------+
|  auth_key  |           b2c8e585cbc68a80           |
| created_at |      2013-10-14T15:20:10.121458      |
|     id     | 6e4e9aa4-bed5-4f94-8f76-df43232f44dc |
|    name    |                 None                 |
| volume_id  | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |
+------------+--------------------------------------+
```

> 注意：
> 您可以使用 `--display-name displayName` 来指定一次转移的代号。
>
> 注意：
> 虽然 `auth_key` 这一项在您执行 `cinder transfer-create VOLUME_ID` 时能显示出来，但是在执行 `cinder transfer-show TRANSFER_ID` 时是显示不出来的。

1. 将卷转移ID和授权码发给受让人（比如用邮件发给他）
2. 查看正在进行中的转移

```
$ cinder transfer-list
+--------------------------------------+---------------------------
|                  ID                  |              VolumeID
+--------------------------------------+---------------------------
| 6e4e9aa4-bed5-4f94-8f76-df43232f44dc | a1cdace0-08e4-4dc7-b9dc-45
+--------------------------------------+---------------------------
```

1. 在卷的受让人，或者新主人，接受了这次转移，您会发现本次转移已经不在转移列表上了

```
$ cinder transfer-list
+----+-----------+------+
| ID | Volume ID | Name |
+----+-----------+------+
+----+-----------+------+
```

接受卷转移请求

1. 如果您是卷的接受者，您要从卷的原主人处获取到transfer ID和授权码。
2. 接受此次转移请求。

```
$ cinder transfer-accept transferID authKey
```

例如：

```
$ cinder transfer-accept 6e4e9aa4-bed5-4f94-8f76-df43232f44dc   b2c
+-----------+--------------------------------------+
|  Property |                Value                 |
+-----------+--------------------------------------+
|     id    | 6e4e9aa4-bed5-4f94-8f76-df43232f44dc |
|    name   |                 None                 |
| volume_id | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |
+-----------+--------------------------------------+
```

1. 如果您没有足够的剩余空间来接受此次转移，此次转移会被自动拒绝。

删除一次转移请求

1. 列出可用卷及其状态

```
$ cinder list
+-------------+----------------+-------------+------+-----------
|     ID      |     Status     | Display Name | Size | Volume Type
+-------------+----------------+-------------+------+-----------
| 72bfce9f... |     error      |    None     |  1   |    None
| a1cdace0... |awaiting-transfer|    None     |  1   |    None
+-------------+----------------+-------------+------+-----------
```

1. 找到对应的转移ID：

```
$ cinder transfer-list
+--------------------------------------+-------------------------
|                  ID                  |          VolumeID
+--------------------------------------+-------------------------
| a6da6888-7cdf-4291-9c08-8c1f22426b8a | a1cdace0-08e4-4dc7-b9dc-45
+--------------------------------------+-------------------------
```

1. 删除卷的转移操作：

```
$ cinder transfer-delete transferID
```

比如：

```
$ cinder transfer-delete a6da6888-7cdf-4291-9c08-8c1f22426b8a
```

1. 查看卷转移列表，您会发现现在该表已被清空，该卷又可以被转移了。

```
$ cinder transfer-list
+----+-----------+------+
| ID | Volume ID | Name |
+----+-----------+------+
+----+-----------+------+
```

```
$ cinder list
+-----------------+-----------+-------------+------+-------------
|       ID        |   Status  | Display Name | Size | Volume Type
+-----------------+-----------+-------------+------+-------------
| 72bfce9f-ca...  |   error   |     None     |  1   |     None
| a1cdace0-08...  | available |     None     |  1   |     None
+-----------------+-----------+-------------+------+-------------
```

管理共享

共享是由文件存储提供的。您可以对实例开放存取共享文件的权限。如要创建共享，您可以使用 `manila` 命令行客户端。

创建共享网络

1. 创建共享网络

```
$ manila share-network-create --name mysharenetwork --description '

+-------------------+--------------------------------------+
| Property          | Value                                |
+-------------------+--------------------------------------+
| name              | mysharenetwork                       |
| segmentation_id   | None                                 |
| created_at        | 2015-08-17T21:13:29.607489           |
| neutron_subnet_id | 8f56d97d-8495-4a5b-8544-9ae4ee9390fc |
| updated_at        | None                                 |
| network_type      | None                                 |
| neutron_net_id    | 394246ed-d3fd-4a30-a456-7042ce3429b9 |
| ip_version        | None                                 |
| nova_net_id       | None                                 |
| cidr              | None                                 |
| project_id        | d80a6323e99f4f22a26ad2accd3ec791     |
| id                | ccd6b453-8b05-4508-bbce-93bfe660451f |
| description       | My Manila network                    |
+-------------------+--------------------------------------+
```

1. 列出共享网络

```
$ manila share-network-list

+--------------------------------------+----------------+
| id                                   | name           |
+--------------------------------------+----------------+
| ccd6b453-8b05-4508-bbce-93bfe660451f | mysharenetwork |
+--------------------------------------+----------------+
```

创建共享

1. 创建一份共享

```
$ manila create --name myshare --description "My Manila share" --sh

+-------------------+--------------------------------------+
| Property          | Value                                |
+-------------------+--------------------------------------+
| status            | creating                             |
| description       | My Manila share                      |
| availability_zone | nova                                 |
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f |
| export_locations  | []                                   |
| host              | None                                 |
| snapshot_id       | None                                 |
| is_public         | False                                |
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7 |
| size              | 1                                    |
| name              | myshare                              |
| share_type        | default                              |
| created_at        | 2015-08-17T21:17:23.777696           |
| export_location   | None                                 |
| share_proto       | NFS                                  |
| project_id        | d80a6323e99f4f22a26ad2accd3ec791     |
| metadata          | {}                                   |
+-------------------+--------------------------------------+
```

1. 显示一份共享的细节

```
$ manila show 2fe736d1-08ac-46f9-a482-8f224405f2a7

+-------------------+--------------------------------------+
| Property          | Value                                |
+-------------------+--------------------------------------+
| status            | creating                             |
| description       | My Manila share                      |
| availability_zone | nova                                 |
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f |
| export_locations  | []                                   |
| host              | ubuntuManila@generic1#GENERIC1       |
| snapshot_id       | None                                 |
| is_public         | False                                |
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7 |
| size              | 1                                    |
| name              | myshare                              |
| share_type        | default                              |
| created_at        | 2015-08-17T21:17:23.000000           |
| export_location   | None                                 |
| share_proto       | NFS                                  |
| project_id        | d80a6323e99f4f22a26ad2accd3ec791     |
| metadata          | {}                                   |
+-------------------+--------------------------------------+
```

1. 列出所有的共享

```
$ manila list

+--------------------------------------+---------+------+----------
| ID                                   | Name    | Size | Share Pro
+--------------------------------------+---------+------+----------
| 2fe736d1-08ac-46f9-a482-8f224405f2a7 | myshare | 1    | NFS
+--------------------------------------+---------+------+----------
```

赋予访问权限

1. 赋予访问权限

```
$ manila access-allow 2fe736d1-08ac-46f9-a482-8f224405f2a7 ip 192.1

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| share_id     | 2fe736d1-08ac-46f9-a482-8f224405f2a7 |
| deleted      | False                                |
| created_at   | 2015-08-17T21:36:52.025125           |
| updated_at   | None                                 |
| access_type  | ip                                   |
| access_to    | 192.100.00.168                       |
| access_level | rw                                   |
| state        | new                                  |
| deleted_at   | None                                 |
| id           | d73d04ca-a97e-42bb-94b1-e01c72c8e50e |
+--------------+--------------------------------------+
```

1. 列出权限列表

```
$ manila access-list 2fe736d1-08ac-46f9-a482-8f224405f2a7

+--------------------------------------+-------------+------------
| id                                   | access type | access to
+--------------------------------------+-------------+------------
| d73d04ca-a97e-42bb-94b1-e01c72c8e50e | ip          | 192.100.00.1
+--------------------------------------+-------------+------------
```

收回访问权限

1. 收回访问权限

```
$ manila access-deny 2fe736d1-08ac-46f9-a482-8f224405f2a7 d73d04ca-
```

1. 列出权限列表

```
$ manila access-list 2fe736d1-08ac-46f9-a482-8f224405f2a7

+----+-------------+-----------+--------------+-------+
| id | access type | access to | access level | state |
+----+-------------+-----------+--------------+-------+
+----+-------------+-----------+--------------+-------+
```

创建快照

1. 创建一份快照

```
$ manila snapshot-create --name mysnapshot --description "My Manila

+-------------+--------------------------------------+
| Property    | Value                                |
+-------------+--------------------------------------+
| status      | creating                             |
| share_id    | 2fe736d1-08ac-46f9-a482-8f224405f2a7 |
| name        | mysnapshot                           |
| created_at  | 2015-08-17T21:50:53.295017           |
| share_proto | NFS                                  |
| id          | 1a411703-baef-495f-8e9c-b60e68f2e657 |
| size        | 1                                    |
| share_size  | 1                                    |
| description | My Manila snapshot                   |
+-------------+--------------------------------------+
```

1. 列出快照列表

```
$ manila snapshot-list

+--------------------------------------+-------------------------
| ID                                   | Share ID
+--------------------------------------+-------------------------
| 1a411703-baef-495f-8e9c-b60e68f2e657 | 2fe736d1-08ac-46f9-a482-8f
+--------------------------------------+-------------------------
```

从快照创建共享

1. 从某份快照创建一个共享

```
$ manila create --snapshot-id 1a411703-baef-495f-8e9c-b60e68f2e657

+-------------------+--------------------------------------+
| Property          | Value                                |
+-------------------+--------------------------------------+
| status            | creating                             |
| description       | None                                 |
| availability_zone | nova                                 |
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f |
| export_locations  | []                                   |
| host              | ubuntuManila@generic1#GENERIC1       |
| snapshot_id       | 1a411703-baef-495f-8e9c-b60e68f2e657 |
| is_public         | False                                |
| id                | bcc5b2a7-862b-418a-9607-5d669619d652 |
| size              | 1                                    |
| name              | mysharefromsnap                      |
| share_type        | default                              |
| created_at        | 2015-08-17T21:54:43.000000           |
| export_location   | None                                 |
| share_proto       | NFS                                  |
| project_id        | d80a6323e99f4f22a26ad2accd3ec791     |
| metadata          | {}                                   |
+-------------------+--------------------------------------+
```

1. 列出共享

```
$ manila list

+--------------------------------------+-----------------+------+--
| ID                                   | Name            | Size | S
+--------------------------------------+-----------------+------+--
| 2fe736d1-08ac-46f9-a482-8f224405f2a7 | myshare         | 1    | N
| bcc5b2a7-862b-418a-9607-5d669619d652 | mysharefromsnap | 1    | N
+--------------------------------------+-----------------+------+--
```

1. 打印从快照创建出的这份共享的详情

```
$ manila show bcc5b2a7-862b-418a-9607-5d669619d652

+-------------------+------------------------------------------
| Property          | Value
+-------------------+------------------------------------------
| status            | available
| description       | None
| availability_zone | nova
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f
| export_locations  | 10.254.0.3:/shares/share-bcc5b2a7-862b-418a-9
| host              | ubuntuManila@generic1#GENERIC1
| snapshot_id       | 1a411703-baef-495f-8e9c-b60e68f2e657
| is_public         | False
| id                | bcc5b2a7-862b-418a-9607-5d669619d652
| size              | 1
| name              | mysharefromsnap
| share_type        | default
| created_at        | 2015-08-17T21:54:43.000000
| share_proto       | NFS
| project_id        | d80a6323e99f4f22a26ad2accd3ec791
| metadata          | {}
+-------------------+------------------------------------------
```

删除共享

1. 删除共享

```
$ manila delete bcc5b2a7-862b-418a-9607-5d669619d652
```

1. 列出共享

```
$ manila list

+--------------------------------------+----------------+------+--
| ID                                   | Name           | Size | S
+--------------------------------------+----------------+------+--
| 2fe736d1-08ac-46f9-a482-8f224405f2a7 | myshare        | 1    | N
| bcc5b2a7-862b-418a-9607-5d669619d652 | mysharefromsnap | 1    | N
+--------------------------------------+----------------+------+--
```

此时共享已删除。

删除快照

1. 在删除快照之前，先列出快照

```
$ manila snapshot-list

+--------------------------------------+---------------------------
| ID                                   | Share ID
+--------------------------------------+---------------------------
| 1a411703-baef-495f-8e9c-b60e68f2e657 | 2fe736d1-08ac-46f9-a482-8f
+--------------------------------------+---------------------------
```

1. 删除快照

```
$ manila snapshot-delete 1a411703-baef-495f-8e9c-b60e68f2e657xyang@
```

1. 再次列出快照列表

```
$ manila snapshot-list

+--------------------------------------+---------------------------
| ID                                   | Share ID
+--------------------------------------+---------------------------
+--------------------------------------+---------------------------
```

此时快照已删除。

为共享盘扩充空间

1. 扩展共享

```
$ manila extend 2fe736d1-08ac-46f9-a482-8f224405f2a7 2
```

1. 查看正在扩展的共享

```
$ manila show 2fe736d1-08ac-46f9-a482-8f224405f2a7

+-------------------+------------------------------------------
| Property          | Value
+-------------------+------------------------------------------
| status            | extending
| description       | My Manila share
| availability_zone | nova
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f
| export_locations  | 10.254.0.3:/shares/share-2fe736d1-08ac-46f9-a
| host              | ubuntuManila@generic1#GENERIC1
| snapshot_id       | None
| is_public         | False
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7
| size              | 1
| name              | myshare
| share_type        | default
| created_at        | 2015-08-17T21:17:23.000000
| share_proto       | NFS
| project_id        | d80a6323e99f4f22a26ad2accd3ec791
| metadata          | {}
+-------------------+------------------------------------------
```

1. 在扩展动作结束后再次查看

```
$ manila show 2fe736d1-08ac-46f9-a482-8f224405f2a7

+-------------------+--------------------------------------------
| Property          | Value
+-------------------+--------------------------------------------
| status            | available
| description       | My Manila share
| availability_zone | nova
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f
| export_locations  | 10.254.0.3:/shares/share-2fe736d1-08ac-46f9-a
| host              | ubuntuManila@generic1#GENERIC1
| snapshot_id       | None
| is_public         | False
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7
| size              | 2
| name              | myshare
| share_type        | default
| created_at        | 2015-08-17T21:17:23.000000
| share_proto       | NFS
| project_id        | d80a6323e99f4f22a26ad2accd3ec791
| metadata          | {}
+-------------------+--------------------------------------------
```

为共享盘缩减容量

1. 缩减共享盘容量

```
$ manila shrink 2fe736d1-08ac-46f9-a482-8f224405f2a7 1
```

1. 在缩减的过程中查看共享盘详情

```
$ manila show 2fe736d1-08ac-46f9-a482-8f224405f2a7

+-------------------+------------------------------------------
| Property          | Value
+-------------------+------------------------------------------
| status            | shrinking
| description       | My Manila share
| availability_zone | nova
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f
| export_locations  | 10.254.0.3:/shares/share-2fe736d1-08ac-46f9-a
| host              | ubuntuManila@generic1#GENERIC1
| snapshot_id       | None
| is_public         | False
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7
| size              | 2
| name              | myshare
| share_type        | default
| created_at        | 2015-08-17T21:17:23.000000
| share_proto       | NFS
| project_id        | d80a6323e99f4f22a26ad2accd3ec791
| metadata          | {}
+-------------------+------------------------------------------
```

1. 在缩减过程结束后再次查看共享盘详情

```
$ manila show 2fe736d1-08ac-46f9-a482-8f224405f2a7

+-------------------+------------------------------------------
| Property          | Value
+-------------------+------------------------------------------
| status            | available
| description       | My Manila share
| availability_zone | nova
| share_network_id  | ccd6b453-8b05-4508-bbce-93bfe660451f
| export_locations  | 10.254.0.3:/shares/share-2fe736d1-08ac-46f9-a
| host              | ubuntuManila@generic1#GENERIC1
| snapshot_id       | None
| is_public         | False
| id                | 2fe736d1-08ac-46f9-a482-8f224405f2a7
| size              | 1
| name              | myshare
| share_type        | default
| created_at        | 2015-08-17T21:17:23.000000
| share_proto       | NFS
| project_id        | d80a6323e99f4f22a26ad2accd3ec791
| metadata          | {}
+-------------------+------------------------------------------
```

为实例设置权限和安全选项

当您启动一个虚拟机时，您可以向其中注入一对密钥，这样您就能通过SSH访问实例了。不过，这样做的前提是您的实例中必须安装了 `cloud-init` 这个包。

您可以为一个实例创建不止一对密钥，您也可以将一对密钥放在同一Project下的多个服务器中。如果您的密钥对是用外部工具创建的，您还可以将其导入进OpenStack。

> 注意：一对密钥只属于一个用户，并不属于一个组。如果要和多个用户共享一对密钥，每个最终用户都要把相同的密钥导入一次。

如果某个镜像是用固定的root密码，或一个静态的密钥来做认证的，您就不能在启动这个实例时创建密钥对了。

“安全组”是一组网络规则，它能限制对实例的网络访问类型。当您启动实例时，您可以为其设置一个或多个安全组。如果您没有创建安全组，新的实例会被自动分配默认的安全组，除非您另外指定别的组了。

安全组里的安全规则控制着组内实例的网络。任何不符合规则的网络流量都会被拒绝。您可以在网络组中添加或删除规则，还可以修改默认安全组或者其他安全组内的规则。

您可以在安全组中设置允许连接实例的哪些端口（或者是通过哪些协议来连接）。例如，您可以修改规则，允许外部通过SSH来连接服务器，Ping这个服务器，或者允许UDP连接（比如DNS之类的服务）。具体说来，您可以指定规则中的如下参数：

```
- 网络流量的来源：可以设置成允许云中的某个IP，其他组的某个IP，或者所有的IP。
- 协议：如果连接SSH的话就要选TCP。如果要ping的话就选ICMP。有时候还可能要选UD
- 虚拟机上的目标端口：定义一个端口范围。如果只想开放一个端口，则需要把这个端口写
```

如果您修改了规则，这些规则即刻生效。

> 注意：使用了默认安全组的实例是不能被任何外部IP访问到的。如果您希望用这些外部IP来访问实例，您必须修改默认安全组中的规则。 您也可以为运行中的实例分配一个浮动IP，这样便能从云的外部访问到了。请参看Manage IP Addresses章节。

添加一对密钥

您可以生成一对密钥，也可以将已经生成的公钥上传进OpenStack

1. 如果想生成一对密钥，请使用如下命令：

```
$ nova keypair-add KEY_NAME > MY_KEY.pem
```

这条命令会生成一对密钥，其中 `KEY_NAME` 是您指定的这对密钥的名称，私钥会保存在您指定的 `.pem` 文件中，然后将公钥注册进Nova数据库里。

1. 运行如下命令，保证只有您能读写这个文件的内容：

```
$ chmod 600 MY_KEY.pem
```

导入一对密钥

1. 如果您已经创建了一对密钥了，假如您的公钥存放在 `~/.ssh/id_rsa.pub` ，您可以运行如下命令来上传公钥。

```
$ nova keypair-add --pub_key ~/.ssh/id_rsa.pub KEY_NAME
```

这条命令会将您的公钥注册在Nova数据库中，保存的名称是您指定的，此处是KEY_NAME。

1. 为了确认您的密钥已经成功导入，用下面的命令列出所有的密钥对：

```
$ nova keypair-list
```

创建和管理安全组

1. 如果要列出当前Project所有的安全组，包括描述，请使用以下命令：

```
$ nova secgroup-list
```

1. 如果要创建一个安全组，请使用如下命令（该命令同时会为安全组命名，还会添加描述）：

```
$ nova secgroup-create SECURITY_GROUP_NAME GROUP_DESCRIPTION
```

1. 如果要删除某个安全组，请使用如下命令：

```
$ nova secgroup-delete SECURITY_GROUP_NAME
```

> 注意：
> 您不能删除某个Project下的默认安全组，也不能删除已经分配给一个运行中的实例的安全组。

创建和管理安全组规则

`nova secgroup-*-rule` 命令专门用来修改用户组规则。在您开始修改这些规则前，请先载入OpenStack RC 文件。详情请参阅Set environment variables using the OpenStack RC file。

````
1. 如果您要列出安全组下地所有规则，请执行以下命令：
```
$ nova secgroup-list-rules SECURITY_GROUP_NAME
```


2. 如果要允许本机的SSH，从以下两个命令中选择一个：


    - 允许所有的网络连接，用CIDR格式写明0.0.0.0/0IP和子网。
    ```
    $ nova secgroup-add-rule SECURITY_GROUP_NAME tcp 22 22 0.0.0.0/
    ```
    - 只允许别的安全组内的IP来访问特定端口：
    ```
    $ nova secgroup-add-group-rule --ip_proto tcp --from_port 22 \
          --to_port 22 SECURITY_GROUP_NAME SOURCE_GROUP_NAME
    ```
````

```
3. 如果要允许实例接受ping，从以下两个命令中任选一个：

    - 允许从任意地址发来的ping，用CIDR格式写明0.0.0.0/0IP和子网。
    ```
    $ nova secgroup-add-rule SECURITY_GROUP_NAME icmp -1 -1 0.0.0.(
    ```
    这条命令将打开所有ICMP流量的所有编码和种类。

    - 只允许别的安全组内的IP来ping：
    ```
    $ nova secgroup-add-group-rule --ip_proto icmp --from_port -1 \
         --to_port -1 SECURITY_GROUP_NAME SOURCE_GROUP_NAME
    ```

4. 如果要允许UDP通信，比如需要在虚拟机上搭建DNS的时候，请从下面两个命令中任选一

    - 允许从任意地址发来的UDP通信，用CIDR格式写明0.0.0.0/0IP和子网。
    ```
    $ nova secgroup-add-rule SECURITY_GROUP_NAME udp 53 53 0.0.0.0/
    ```
    - 只允许别的安全组内的IP来访问特定端口：
    ```
    $ nova secgroup-add-group-rule --ip_proto udp --from_port 53 \
          --to_port 53 SECURITY_GROUP_NAME SOURCE_GROUP_NAME
    ```
```

删除用户组规则

如果要删除安全组规则，您要在删除指令后面写明您创建这个规则时添加的参数。

比如，如果您要删除允许所有IP连接SSH的规则，请使用如下指令：

```
$ nova secgroup-delete-rule SECURITY_GROUP_NAME tcp 22 22 0.0.0.0/(
```

创建实例

实例（Instance）是运行在云中的虚拟机。

在创建新实例之前，您要提前知晓以下参数：

- 实例的源可以是镜像，快照，或者包含镜像或快照的块存储设备。
- 实例的名字。
- 您实例的型号，这个型号决定了您nova实例的CPU，内存和磁盘空间情况。型号（flavor）是您虚拟机的硬件可用配置。它决定了您能创建的虚拟机的大小。
- 任意的用户数据文件。用户数据文件时在元数据服务中的一个特殊的键，它保存了一份能给虚拟机实例中的云服务使用的文件。比如，`cloud-init` 程序便使用了用户数据文件，这个程序是一个源自Ubuntu的开源包，能用在多个Linux发行版上，能够接管云实例的初始化过程。
- 访问权限和安全认证信息，其中包括以下认证信息：
    - 密钥对。密钥对能在镜像启动的时候注入进镜像中，不过前提时该镜像必须包含cloud-init包。您要为每个Project建立至少一个密钥对。如果您已经通过外部工具创建了密钥对，您可以将其导入OpenStack。在一个Project内的多个实例可以共用一个密钥对。
    - 安全组。用来决定哪些网络流量能流至实例。安全组中包含了一组防火墙策略，被成为安全组规则。
- 如果您需要，您可以为运行中的实例分配一个浮动**IP**（公网地址）。
- 您还可以为实例安排一个块存储设备，或者说卷，来做持久化存储。

> 注意：
> 使用默认安全组的实例，默认情况下是不能为任何外部IP访问到的。如果您想要让外部IP地址访问到某个实例，您必须修改默认安全组的规则。 不过您还可以通过分配浮动IP的方式让外部IP得以访问我们的实例。请参见Manage IP Addresses。

在您已经知晓了上述参数之后，您便可以通过镜像或卷来启动实例了。您可以从OpenStack的镜像直接启动实例，也可以从持久化卷上的镜像启动实例。OpenStack镜像服务提供了一个镜像池，能将其中的镜像提供给不同Project中的用户。

收集启动实例时需要用到的参数

在您开始之前，请先执行OpenStack RC文件。

1. 列出可用的实例型号：

```
$nova flavor-list
```

记下您想使用的服务器型号。

```
+-----+-----------+-----------+------+-----------+------+-------+-
| ID  | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | F
+-----+-----------+-----------+------+-----------+------+-------+-
| 1   | m1.tiny   | 512       | 1    | 0         |      | 1     | 1
| 2   | m1.small  | 2048      | 20   | 0         |      | 1     | 1
| 3   | m1.medium | 4096      | 40   | 0         |      | 2     | 1
| 4   | m1.large  | 8192      | 80   | 0         |      | 4     | 1
| 5   | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1
+-----+-----------+-----------+------+-----------+------+-------+-
```

1. 列出可用镜像：

```
$ nova image-list
```

记下您想使用哪个镜像来启动您的实例。

```
+--------------------------------------+---------------------------
| ID                                   | Name
+--------------------------------------+---------------------------
| 397e713c-b95b-4186-ad46-6126863ea0a9 | cirros-0.3.2-x86_64-uec
| df430cc2-3406-4061-b635-a51c16e488ac | cirros-0.3.2-x86_64-uec-ke
| 3cf852bd-2332-48f4-9ae4-7d926d50945e | cirros-0.3.2-x86_64-uec-ra
+--------------------------------------+---------------------------
```

您也可以用 `grep` 来过滤结果，像下面这样：

```
$ nova image-list | grep 'kernel'

| df430cc2-3406-4061-b635-a51c16e488ac | cirros-0.3.2-x86_64-uec-ke
```

1. 列出可用的安全组：

```
$ nova secgroup-list --all-tenants
```

> 注意：
> 如果您是管理员，您可以指定 `--all-tenants` 参数来列出所有tenant的安全组。

记下您想套用在您新实例上的安全组。

```
+----+---------+-------------+----------------------------------+
| Id | Name    | Description | Tenant_ID                        |
+----+---------+-------------+----------------------------------+
| 2  | default | default     | 66265572db174a7aa66eba661f58eb9e |
| 1  | default | default     | b70d90d65e464582b6b2161cf3603ced |
+----+---------+-------------+----------------------------------+
```

如果您没有创建任何安全组，您可以将实例设置为只使用默认安全组。

您还可以用下面的命令来查看某一个安全组内的详细规则：

```
$ nova secgroup-list-rules default
```

1. 列出可用的密钥对，记下您想用在新实例SSH上的那对密钥。

```
$ nova keypair-list
```

通过镜像创建实例

1. 在您已经掌握了所需的参数，运行如下命令来启动实例。要指定服务器名，型

号ID和镜像ID：

```
$ nova boot --flavor FLAVOR_ID --image IMAGE_ID --key-name kEY_NAME
-- user-data USER_DATA_FILE --security-groups SEC_GROUP_NAME --meta
INSTANCE_NAME
```

此外，您可以在上面的命令中传入一个 `--key-name` 参数以指定您要用哪一个密钥对，以及用 `--security-groups` 参数来指定您要用哪一个安全组。您还可以在参数中添加元数据键值对，比如，您如果要为您的服务器添加一段简短的描述，用 `--meta description="My Server"` 参数即可。

您还可以用 `--user-data USER-DATA-FILE` 将用户数据文件传入新建的实例。

> 重要： 如果您创建实例时， `INSTANCE_NAME` 这一项的长度超过了63个字符，Compute会将其截断，保证dnsmasq能正常工作。这一动作的日志同时会保存在 `nova-network.log` 中。

请参看下面的命令。该命令创建了一个叫myCirrosServer的实例，型号时 `m1.small` （ID是1），镜像是cirros-0.3.2-x86_64-uec（ID是397e713c-b95b-4186-ad46-6126863ea0a9)，默认用户组，密钥对是 `KeyPair01` ，用户数据文件是 `cloudinit.file` 。

```
$ nova boot --flavor 1 --image 397e713c-b95b-4186-ad46-6126863ea0a9
--security-groups default --key-name KeyPair01 --user-data cloudini
myCirrosServer
```

根据您提供的参数，上面的命令会返回如下服务器属性：

```
+--------------------------------------+-----------------------------
| Property                             | Value
+--------------------------------------+-----------------------------
| OS-EXT-STS:task_state                | scheduling
| image                                | cirros-0.3.2-x86_64-uec
| OS-EXT-STS:vm_state                  | building
| OS-EXT-SRV-ATTR:instance_name        | instance-00000002
| flavor                               | m1.small
| id                                   | b3cdc6c0-85a7-4904-ae85-71
| security_groups                      | [{u'name': u'default'}]
| user_id                              | 376744b5910b4b4da7d8e6cb48
| OS-DCF:diskConfig                    | MANUAL
| accessIPv4                           |
| accessIPv6                           |
| progress                             | 0
| OS-EXT-STS:power_state               | 0
| OS-EXT-AZ:availability_zone          | nova
| config_drive                         |
| status                               | BUILD
| updated                              | 2013-07-16T16:25:34Z
| hostId                               |
| OS-EXT-SRV-ATTR:host                 | None
| key_name                             | KeyPair01
| OS-EXT-SRV-ATTR:hypervisor_hostname  | None
| name                                 | myCirrosServer
| adminPass                            | tVs5pL8HcPGw
| tenant_id                            | 66265572db174a7aa66eba661f
| created                              | 2013-07-16T16:25:34Z
| metadata                             | {u'KEY': u'VALUE'}
+--------------------------------------+-----------------------------
```

如果状态是“BUILD”，意味着实例已启动，但尚未上线。

如果状态是“ACTIVE”，意味着实例已经可以用了。

1. 在打印出来的参数中查找ID这一项，这一项是服务器ID。您可以用这个ID获取到服务器详情，或者删掉这个服务器。

2. 在参数中查找adminPass项，将管理员密码复制出来。用这个密码来登录服务器。

> 注意： 您可以在创建实例时，将任意本地文件注入系统，用 `--file <dst-path=src-path>` 即可。您最多能存入五个文件。比如，您想用自定义的认证密钥文件放在实例中，而不是使用正常的SSH密钥注入的话，您可以用 `--file` 选项。如下是代码示例：
>
> ```
> $ nova boot --image ubuntu-cloudimage --flavor 1 vn-name\
> --file /root/.ssh/authorized_keys=special_authorized_keysfile
> ```

1. 检查一下实例是否已上线成功：

```
$ nova list
```

该列表显示您在这个Project下的服务器ID，名称，状态，私网IP（如果有公网IP也会显示）。

```
+-------------+---------------------+--------+------------+-------
| ID          | Name                | Status | Task State | Power 
+-------------+---------------------+--------+------------+-------
| 84c6e57d... | myCirrosServer      | ACTIVE | None       | Runnir
| 8a99547e... | myInstanceFromVolume | ACTIVE | None       | Runnir
+-------------+---------------------+--------+------------+-------
```

如果实例的状态是 `ACTIVE` ，则该实例已经上线了。

1. 如果要查看所有 `nova list` 后面可以接的命令，可以执行以下命令：

```
$ nova help list
```

> 注意： 如果您不提供密钥对，安全组，或者安全组规则，那您只能通过云内提供的VNC来访问这个实例。连ping这个实例都ping不通。

管理实例和主机

“实例”，是运行在云中物理主机上的虚拟机。Compute服务管理着众多的实例。“主机”是一组实例栖身的物理节点。

本章介绍了实例管理中的各种不同的任务，比如添加浮动IP，关闭和启动实例，以及删除实例等等。本节也介绍了一些物理节点的管理任务。

管理IP地址

每个实例都有一个内网的固定IP，不过还可以为其添加一个对外的IP，或者叫“浮动IP”。内网IP是用来和其他实例间通信的，而公网是用来和外网的云设施通信的，而且也经常用来和互联网上的设备通信。

当您创建了一个实例，OpenStack会自动为其分配一个内网IP，这个地址一直会保持不变，直到您将该实例销毁。您重启实例也不会影响这个内网IP。

浮动IP资源池则是由云管理员设置的，该服务在OpenStack Compute中可用。每个Project的资源配额决定了您能在这个Project分配的浮动IP的最大数量。在您给某个实例分配了一个浮动IP之后，您能够：

- 将该浮动IP地址和Project中的这个实例相关联。但是，一个浮动IP，同一时间内只能分配给一个实例。
- 将该浮动IP从实例上解除关联。
- 从Project中删除浮动IP，并且连带删除该IP的各种关联。

您可以使用 `nova floating-ip-*` 命令来管理浮动IP地址。

列出浮动**IP**信息

列出所有的浮动IP资源池，请使用如下命令：

```
$ nova floating-ip-pool-list
+--------+
| name   |
+--------+
| public |
| test   |
+--------+
```

> 注意： 如果这个列表里没有东西，云管理员必须先创建一个浮动IP资源池。

想列出分配给当前Project的所有浮动IP地址，请使用如下命令：

```
$ nova floating-ip-list
+--------------+--------------------------------------+----------+-
| Ip           | Instance Id                          | Fixed Ip |
+--------------+--------------------------------------+----------+-
| 172.24.4.225 | 4a60ff6a-7a3c-49d7-9515-86ae501044c6 | 10.0.0.2 |
| 172.24.4.226 | None                                 | None     |
+--------------+--------------------------------------+----------+-
```

对于每个分配给当前Project的浮动IP，上面的这个命令能返回以下四个结果：该浮动IP地址，该浮动IP关联的实例的ID，关联的固定IP地址，以及这个浮动IP地址的来源池。

分配浮动**IP**

您可以将浮动IP分配给Project，也可以分配给instance。

1. 运行如下命令，将一个浮动IP分配给当前Project。默认情况下，该浮动IP是从公共的资源池而来的。该命令的返回结果中会显示本次分配的地址。

```
$ nova floating-ip-create
+--------------+-------------+----------+--------+
| Ip           | Instance Id | Fixed Ip | Pool   |
+--------------+-------------+----------+--------+
| 172.24.4.225 | None        | None     | public |
+--------------+-------------+----------+--------+
```

> 注意： 如果IP地址资源池不止有一个，您可以将资源池名称作为参数传递给该命令，来制定要使用哪一个资源池。比如，要从 `test` 池中抽取IP地址，请使用如下命令：
>
> ```
> $ nova floating-ip-create test
> ```

1. 列出该Project下所有可以关联浮动IP地址的实例：

```
$ nova list
+---------------------+------+---------+------------+--------------
| ID                  | Name | Status  | Task State | Power State
+---------------------+------+---------+------------+--------------
| d5c854f9-d3e5-4f... | VM1  | ACTIVE  | -          | Running
| 42290b01-0968-43... | VM2  | SHUTOFF | -          | Shutdown
+---------------------+------+---------+------------+--------------
```

1. 用如下命令，将IP地址和实例相关联：

```
$ nova floating-ip-associate INSTANCE_NAME_OR_ID FLOATING_IP_ADDRES
```

比如像下面这样：

```
$ nova floating-ip-associate VM1 172.24.4.225
```

此时，该实例便已经和两个IP地址绑定了。

```
$ nova list
+------------------+------+--------+------------+-------------+---
| ID               | Name | Status | Task State | Power State | Net
+------------------+------+--------+------------+-------------+---
| d5c854f9-d3e5... | VM1  | ACTIVE | -          | Running     | pri
| 42290b01-0968... | VM2  | SHUTOFF| -          | Shutdown    | pri
+------------------+------+--------+------------+-------------+---
```

您在关联了该IP地址，且为实例配置了安全组规则之后，该实例便能通过浮动IP来访问了。

> 注意： 如果某个实例连接了多个网络，用 `--fixed-address` 即可将浮动IP和一个固定IP关联。
>
> ```
> $ nova floating-ip-associate --fixed-address FIXED_IP_ADDRESS \
>  INSTANCE_NAME_OR_ID FLOATING_IP_ADDRESS
> ```

回收浮动IP

若要解除浮动IP和实例的关联，请使用如下命令：

```
$ nova floating-ip-disassociate INSTANCE_NAME_OR_ID FLOATING_IP_ADD
```

若要解除浮动IP和Project，请使用如下命令：

```
$ nova floating-ip-delete FLOATING_IP_ADDRESS
```

此时，该IP地址会回归地址资源池，可以被所有Project使用。如果您直接将某一个浮动IP和Project解除了关联，且正挂在某个运行的实例上时，该IP同样会被自动收回。

调整服务器大小

通过调整型号来调整服务器大小。

1. 请首先查看服务器信息，包括服务器的型号大小，这里的型号大小是体现在 `flavor` 上的：

```
$ nova show myCirrosServer
+-------------------------------------+---------------------------
| Property                            | Value
+-------------------------------------+---------------------------
| status                              | ACTIVE
| updated                             | 2013-07-18T15:08:20Z
| OS-EXT-STS:task_state               | None
| OS-EXT-SRV-ATTR:host                | devstack
| key_name                            | None
| image                               | cirros-0.3.2-x86_64-uec (39
| private network                     | 10.0.0.3
| hostId                              | 6e1e69b71ac9b1e6871f91e2dfc
| OS-EXT-STS:vm_state                 | active
| OS-EXT-SRV-ATTR:instance_name       | instance-00000005
| OS-EXT-SRV-ATTR:hypervisor_hostname | devstack
| flavor                              | m1.small (2)
| id                                  | 84c6e57d-a6b1-44b6-81eb-fcb
| security_groups                     | [{u'name': u'default'}]
| user_id                             | 376744b5910b4b4da7d8e6cb483
| name                                | myCirrosServer
| created                             | 2013-07-18T15:07:59Z
| tenant_id                           | 66265572db174a7aa66eba661f5
| OS-DCF:diskConfig                   | MANUAL
| metadata                            | {u'description': u'Small te
| accessIPv4                          |
| accessIPv6                          |
| progress                            | 0
| OS-EXT-STS:power_state              | 1
| OS-EXT-AZ:availability_zone         | nova
| config_drive                        |
+-------------------------------------+---------------------------
```

如上显示，该服务器的型号是 `m1.small(2)`

1. 用如下的方法列出可用的型号

```
$ nova flavor-list
+-----+-----------+-----------+------+-----------+------+-------+--
| ID  | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | R
+-----+-----------+-----------+------+-----------+------+-------+--
| 1   | m1.tiny   | 512       | 1    | 0         |      | 1     | 1
| 2   | m1.small  | 2048      | 20   | 0         |      | 1     | 1
| 3   | m1.medium | 4096      | 40   | 0         |      | 2     | 1
| 4   | m1.large  | 8192      | 80   | 0         |      | 4     | 1
| 5   | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1
+-----+-----------+-----------+------+-----------+------+-------+--
```

1. 如果您希望重新调整服务器大小，请使用 `nova resize` 命令，后面加上服务器ID或者名称，以及新的型号。想要即时查看容量调整过程，请使用 `--poll` 参数。例如：

```
$ nova resize myCirrosServer 4 --poll

Instance resizing... 100% complete
Finished
```

> 注意： 默认情况下， `nova resize` 这个命令会给服务器实例自主关机的时间，此后才会执行关闭电源和调整大小的动作。关机的行为是在 `nova.conf` 中的 `shutdown_timeout` 参数下配置的。这个值表示OpenStack预留给服务器实例执行关机操作的总时间，单位为秒。默认情况下这个时间是60秒。详情请参看Description of Compute configuration options。

1. 查看服务器状态

```
$ nova list
+----------------------+----------------+--------+-----------------
| ID                   | Name           | Status | Networks
+----------------------+----------------+--------+-----------------
| 84c6e57d-a6b1-44b... | myCirrosServer | RESIZE | private=172.16.1
+----------------------+----------------+--------+-----------------
```

容量调整完成后，这里的状态将变为 `VERIFY_RESIZE` 。

1. 确认容量调整完成：

```
$ nova resize-confirm 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5
```

此时服务器状态应为 `ACTIVE` 。

1. 如果容量调整未能按预期完成，您可以回滚调整操作：

```
$ nova resize-revert 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5
```

此时服务器状态也应为 `ACTIVE` 。

关闭和开启实例

您可以使用下面的几种方式来关闭和开启实例。

暂停和恢复实例

如果要暂停实例，请使用如下命令：

```
$ nova pause INSTANCE_NAME
```

该命令会把服务器实例的状态存储在内存中，暂停了的实力会依然运行，只是如同被封冻一样，不会有任何动作。想要恢复实例，请使用如下命令：

```
$ nova unpause INSTANCE_NAME
```

停用和恢复实例

如果要在虚拟化软件层面上将某个实例停用，请使用如下命令：

```
$nova suspend INSTANCE_NAME
```

如果要恢复一个被停用的实例，请使用如下命令：

```
$nova resume INSTANCE_NAME
```

雪藏和恢复实例

如果您有一个暂时不用的实例，但还想让它留在您的服务器列表上，您便可能需要雪藏的功能了。例如，您可以在一周结束时雪藏一个实例，然后在下周一重新打开这个实力。所有相关的数据和资源会被保留，但雪藏时内存的东西是不会保存的。如果一个被雪藏的实例再也没用了，也可以被彻底删除。

您可以执行如下几种有关雪藏的任务：

- 雪藏实例：关闭服务器实例，将关联数据和资源保存（如果没有volume backed，将会创建一个快照）。内存中的信息将会丢失。

```
$ nova shelve SERVERNAME
```

> 注意： 默认情况下， `nova shelve` 这个命令会给服务器实例自主关机的时间，此后才会执行关闭电源和调整大小的动作。关机的行为是在 `nova.conf` 中的 `shutdown_timeout` 参数下配置的。这个值表示OpenStack预留给服务器实例执行关机操作的总时间，单位为秒。默认情况下这个时间是60秒。详情请参看Description of Compute configuration options。关机时间也可以被镜像中的元数据配置覆盖掉。镜像元数据中的 `os_shutdown_timeout` 参数表示该系统需要多长时间才能完成一次关机操作，该参数的优先级要高于 `nova.conf` 中的配置。

- 恢复被雪藏的实例：

```
$ nova unshelve SERVERNAME
```

- 删除被雪藏的实例：将该实例删除，删掉所有的数据，回收资源。如果您不再需要某个实例，您可以将该实例从云中抹除，以节省空间。

```
$ nova shelve-offload SERVERNAME
```

通过IP地址查找实例

在使用 `nova list` 时，您可以用IP地址来查找实例，只需添加 `--ip` 即可。

```
$ nova list --ip IP_ADDRESS
```

如下示例展示了如何查询IP为 `10.0.0.4` 的实例：

```
$ nova list --ip 10.0.0.4
+------------------+---------------------+--------+------------+--
| ID               | Name                | Status | Task State | P
+------------------+---------------------+--------+------------+--
| 8a99547e-7385... | myInstanceFromVolume | ACTIVE | None       | R
+------------------+---------------------+--------+------------+--
```

重启实例

重启实例有两种，分别被称为“软重启”和“硬重启”。软重启会尝试正常关机并重启实例，硬重启会直接将实例“断电”并重启。

默认情况下，如果您通过 `nova` 重启，执行的是软重启。

```
$ nova reboot SERVER
```

如果您需要执行硬重启，添加 `--hard` 参数即可：

```
$ nova reboot --hard SERVER
```

您在重启时还可以进入恢复模式。例如，在您的实例使用时间过长之后，文件系统有损坏时，您可能就需要进入恢复模式了。

> 注意： 实例运行在“恢复模式”下时，暂停、停用和关闭操作是不可用的。因为这些操作会使实例的原始状态丢失，并且无法退出“恢复模式”。

“恢复模式”提供了一种访问实例的机制，这种机制在实例不可访问时依然有效。默认情况下，“恢复模式”会使用初始的景象启动一个实例，然后将当前的启动盘作为第二启动盘挂载到这个实例上。

如果要在“恢复模式”下启动某个实例，请使用如下命令：

```
$ nova rescue SERVER
```

> 注意： 默认情况下， `nova rescue` 这个命令会给服务器实例自主关机的时间，此后才会执行关闭电源和调整大小的动作。关机的行为是在 `nova.conf` 中的 `shutdown_timeout` 参数下配置的。这个值表示OpenStack预留给服务器实例执行关机操作的总时间，单位为秒。默认情况下这个时间是60秒。详情请参看Description of Compute configuration options。关机时间也可以被镜像中的元数据配置覆盖掉。镜像元数据中的 `os_shutdown_timeout` 参数表示该系统需要多长时间才能完成一次关机操作，该参数的优先级要高于 `nova.conf` 中的配置。

如果要在正常模式下重启这个实例，请执行如下命令：

```
$ nova unrescue SERVER
```

如果您在“恢复模式”下启动实例时希望指定一个镜像，而不是用默认的那个镜像，请使用 `--rescue_image_ref` 参数：

```
$ nova rescue --rescue_image_ref IMAGE_ID SERVER
```

删除实例

如果您不再需要某个实例，您可以删了它。

1. 列出所有实例：

```
$ nova list
+------------+---------------------+--------+------------+------
| ID         | Name                | Status | Task State | Power
+------------+---------------------+--------+------------+------
| 84c6e57d... | myCirrosServer      | ACTIVE | None       | Runnir
| 8a99547e... | myInstanceFromVolume | ACTIVE | None       | Runnir
| d7efd3e4... | newServer           | ERROR  | None       | NOSTAT
+------------+---------------------+--------+------------+------
```

1. 运行 `nova delete` 命令来删除实例。下面的例子便是删除一个实例的做法，其中 `newServer` 实例的状态是 `ERROR` 。

```
$ nova delete newServer
```

如果您删除实例成功，是不会有提示的。

1. 如果您想确认服务器的确被删掉了，再执行一次 `nova list` 即可。

```
$ nova list
+-------------+----------------------+--------+------------+-------
| ID          | Name                 | Status | Task State | Power
+-------------+----------------------+--------+------------+-------
| 84c6e57d... | myCirrosServer       | ACTIVE | None       | Runnir
| 8a99547e... | myInstanceFromVolume | ACTIVE | None       | Runnir
+-------------+----------------------+--------+------------+-------
```

被删除的实例已经不会显示在列表中了。

通过控制台连接实例

如果要使用控制台连接实例，我们需要用到VNC或者SPICE。这两种都不在乎控制台日志是否有输出。控制台的方式可以将鼠标和键盘的信号通过中继发送到实例上。

OpenStack中，目前支持了三种远程连接控制台的方式：

novnc：一个通过HTML 5 Canvas和WebSockets实现的浏览器上的VNC客户端。 **spice**：一个运行在浏览器上的连接虚拟化实例的客户端。 **xvpvnc**：Java写成的连接服务器实例的控制端。

例如：

如果需要通过控制台访问实例，请执行如下命令：

```
$ nova get-vnc-console INSTANCE_NAME VNC_TYPE
```

该命令会返回一个URL，通过这个URL，您便可以访问控制台了。

```
+--------+------------------------------------------------------------
| Type   | Url
+--------+------------------------------------------------------------
| xvpvnc | http://192.168.5.96:6081/console?token=c83ae3a3-15c4-48
+--------+------------------------------------------------------------
```

上面的指令中， `VNC_TYPE` 可以是前述三种连接方式的任意一种。

在使用SPICE方式时，您可以直接在实例的页面使用浏览器插件来使用，也可以用 `get-vnc-console` 命令，获取到一个带认证信息的URL来访问使用。上面的例子便是后一种。

欲了解更多信息，或者这三者的对比（包括安全性考虑），请参看Security Guide。

管理裸机（bare-metal）节点

OpenStack Compute的裸机驱动，能通过一些公用的云API，或Orchestration（Heat）来管理硬件主机设备。这个驱动的用例主要是一些单租户的云平台，例如高性能计算平台，或是部署OpenStack本身的时候。

如果您使用裸机驱动，您必须创建一个网络接口，并且将其加在裸机节点上。然后您便可以通过裸机镜像来启动实例了。

您可以列出和删除裸机节点。当您删除一个节点时，与之关联的网络接口都将会被删除。您也可以列出或移除和某个裸机节点相关联的网络接口。

常用指令

管理裸机节点时，我们一般使用如下命令：

`baremetal-interface-add` 将某个网络接口添加在裸机节点上。

`baremetal-interface-list` 列出和某个裸机节点关联的网络接口。

`baremetal-node-create` 创建一个裸机节点。

`baremetal-node-delete` 删除一个裸机节点，同时删除和它关联的所有接口。

`baremetal-node-list` 列出所有可用的裸机节点。

`baremetal-node-show` 列出某一裸机节点的信息。

创建一个**bare-metal**节点

在您创建裸机节点时，您提供的PM地址，用户名和密码应该匹配您硬件设备中BIOS或IPMI上的配置。

```
$ nova baremetal-node-create --pm_address PM_ADDRESS --pm_user PM_U
  --pm_password PM_PASSWORD $(hostname -f) 1 512 10 aa:bb:cc:dd:ee:
```

如下示例展示了创建节点的命令，其中PM地址是 `1.2.3.4` ，PM用户名是 `ipmi` ，密码是 `ipmi` 。

```
$ nova baremetal-node-create --pm_address 1.2.3.4 --pm_user ipmi \
  --pm_password ipmi $(hostname -f) 1 512 10 aa:bb:cc:dd:ee:ff
+------------------+-------------------+
| Property         | Value             |
+------------------+-------------------+
| instance_uuid    | None              |
| pm_address       | 1.2.3.4           |
| interfaces       | []                |
| prov_vlan_id     | None              |
| cpus             | 1                 |
| memory_mb        | 512               |
| prov_mac_address | aa:bb:cc:dd:ee:ff |
| service_host     | ubuntu            |
| local_gb         | 10                |
| id               | 1                 |
| pm_user          | ipmi              |
| terminal_port    | None              |
+------------------+-------------------+
```

在该节点上添加网卡口

对于该节点上的每一个网卡，您都必须创建一个接口，并且指定该接口的MAC地址。

```
$ nova baremetal-interface-add 1 aa:bb:cc:dd:ee:ff
+-------------+-------------------+
| Property    | Value             |
+-------------+-------------------+
| datapath_id | 0                 |
| id          | 1                 |
| port_no     | 0                 |
| address     | aa:bb:cc:dd:ee:ff |
+-------------+-------------------+
```

从一份bare-metal镜像创建实例

“裸机实例”是直接创建在物理机上的虚拟机实例，其下没有任何虚拟层。Nova通过IPMI来管理其电源。有些时候，Nova会通过Neutron和OpenFlow来管理网络。

```
 $ nova boot --image my-baremetal-image --flavor my-baremetal-flavo
 +-----------------------------+-----------------------------------
 | Property                    | Value
 +-----------------------------+-----------------------------------
 | status                      | BUILD
 | id                          | cc302a8f-cd81-484b-89a8-b75eb3911b
 +-----------------------------+-----------------------------------

... wait for instance to become active ...
```

> 注意： 通过 `--availability_zone` 来指定在那个区，或哪个节点上启动服务器。并用逗号将区和主机名分开。
>
> ```
> $ nova boot --availability_zone zone:HOST,NODE
> ```
>
> `host` （主机名）是可选的。您可以只写 `zone:,node` ，但记得要加逗号。

列出bare-metal节点和接口

用 `nova baremetal-node-list` 来查看所有的裸机设备和接口。如果节点正在使用中，这个信息表中会同时显示实例的UUID。

查看bare-metal节点的详细信息

用 `nova baremetal-node-show` 命令来查看某个裸机设备的详情：

```
$ nova baremetal-node-show 1
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| instance_uuid    | cc302a8f-cd81-484b-89a8-b75eb3911b1b |
| pm_address       | 1.2.3.4                              |
| interfaces       | [{u'datapath_id': u'0', u'id': 1,    |
|                  | u'port_no': 0,                       |
|                  | u'address': u'aa:bb:cc:dd:ee:ff'}]   |
| prov_vlan_id     | None                                 |
| cpus             | 1                                    |
| memory_mb        | 512                                  |
| prov_mac_address | aa:bb:cc:dd:ee:ff                    |
| service_host     | ubuntu                               |
| local_gb         | 10                                   |
| id               | 1                                    |
| pm_user          | ipmi                                 |
| terminal_port    | None                                 |
+------------------+--------------------------------------+
```

为实例提供用户数据

用户数据文件时在元数据服务中的一个特殊的键，它保存了一份能给虚拟机实例中的云服务使用的文件。比如，`cloud-init`程序便使用了用户数据文件，这个程序是一个源自Ubuntu的开源包，能用在多个Linux发行版上，它可以接管云实例的初始化过程。

您可以将用户数据写在一份本地文件中，然后在创建实例时用`--user-data <user-data-file>`参数将其传入。

```
$ nova boot --image ubuntu-cloudimage --flavor 1 --user-data mydata
```

利用快照迁移实例

如果要用快照将实例从OpenStack Project迁移到云中，请使用如下方法。

在源Project中：

1. 创建实例的快照
2. 将快照镜像下载下来

在目的Project中：

1. 将快照导入到新的环境中
2. 用这个快照启动新实例

> 注意： 有些云提供商只允许管理员来进行这项操作。

创建实例的快照

1. 关闭您想要迁移的实例，确保在创建快照的时候所有的数据都已保存在硬盘中。如果有必要，您可以列出所有实例，来查看您想要迁移的那个实例的实例名。

```
$ nova list
+--------------------------------------+------------+--------+-----
| ID                                   | Name       | Status | Netw
+--------------------------------------+------------+--------+-----
| c41f3074-c82a-4837-8673-fa7e9fea7e11 | myInstance | ACTIVE | priv
+--------------------------------------+------------+--------+-----

$ nova stop example
```

1. 用 `nova list` 再列出实例，确定您要迁移的实例已经是 `SHUTOFF` 的状态了：

```
$ nova list
+--------------------------------------+------------+---------+----
| ID                                   | Name       | Status  | Net
+--------------------------------------+------------+---------+----
| c41f3074-c82a-4837-8673-fa7e9fea7e11 | myInstance | SHUTOFF | pri
+--------------------------------------+------------+---------+----
```

1. 用 `nova image-create` 命令来创建新快照：

```
$ nova image-create --poll myInstance myInstanceSnapshot
Instance snapshotting... 50% complete
```

1. 用 `nova image-list` 命令检查镜像的状态，直到状态变为 `ACTIVE` ：

```
$ nova image-list
+--------------------------------------+---------------------------
| ID                                   | Name
+--------------------------------------+---------------------------
| 657ebb01-6fae-47dc-986a-e49c4dd8c433 | cirros-0.3.2-x86_64-uec
| 72074c6d-bf52-4a56-a61c-02a17bf3819b | cirros-0.3.2-x86_64-uec-ke
| 3c5e5f06-637b-413e-90f6-ca7ed015ec9e | cirros-0.3.2-x86_64-uec-ra
| f30b204e-1ce6-40e7-b8d9-b353d4d84e7d | myInstanceSnapshot
+--------------------------------------+---------------------------
```

将这个快照镜像下载下来

1. 找到镜像的ID

```
$ nova image-list
+-------------------+-------------------+--------+-----------------
| ID                | Name              | Status | Server
+-------------------+-------------------+--------+-----------------
| f30b204e-1ce6...  | myInstanceSnapshot| ACTIVE | c41f3074-c82a-483
+------------------+-------------------+--------+-----------------
```

1. 用上一步查询到的镜像ID来下载镜像

```
$ glance image-download --file snapshot.raw
f30b204e-1ce6-40e7-b8d9-b353d4d84e7d
```

> 注意： `glance image-download` 这个命令要求必须使用镜像ID，不能使用镜像名。此外，您还要确保您的目标文件夹有足够的空间来存储这个镜像文件。

1. 将该镜像转移到新环境中，用HTTP或者直接上传（scp），任何方法都可以。

将这个快照导入到新系统中

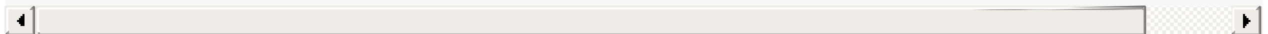
在新的Project或者云环境中，导入快照：

```
$ glance image-create --copy-from IMAGE_URL
```

用新的快照启动实例

在新的Project或者云环境中，用新的快照启动实例：

```
$ nova boot --flavor m1.tiny --image myInstanceSnapshot myNewInstar
```



将元数据存储在配置盘中

您可以让OpenStack将元数据写入到一个特殊的“配置盘”中，实例启动时可以挂载这个盘来读取配置信息。一般情况下，这些信息是由元数据服务提供的。注意，这里的元数据和用户数据不同。

这个功能的一个用例是在没有DHCP的情况下给实例分配IP地址。例如，您可以通过配置盘将IP地址配置传给实例，实例挂载上这个盘以后，读取其中的IP信息，before you configure the network settings for the instance.

使用本功能的要求，和本功能的几个注意事项

要使用配置盘，您的主机和镜像必须满足以下要求。

对主机的要求：

- 以下虚拟机管理器支持配置盘：libvirt，XenServer，Hyper-V和VMware。
- 如果要在libvirt，XenServer或者VMware中使用配置盘，您必须先在每个compute host上安装 `genisoimage` 。否则，实例无法正常启动。 您需要用 `mkisofs_cmd` 来标记您安装genisoimage程序的路径。如果genisoimage和 `nova-compute` 服务在一个路径下，就不用设置了。
- 如果您要在Hyper-V下使用配置盘，您必须用 `mkisofs_cmd` 来标记您安装的mkisofs.exe的全路径。此外，您还要在 `hyperv` 中设置 `qemu_img_cmd` 值，将其指向 `qemu-img` 这个命令的安装位置。

对镜像的要求：

- 带有新版本cloud-init包的镜像能够自动获取到配置盘中的元数据。0.7.1版本的cloud-init包能用在Ubuntu和Fedora系的操作系统上，比如说RHEL。
- 如果镜像中没有安装cloud-init包，您必须定制一下这个镜像：写一个能执行各种动作的脚本，比如在启动的时候挂载配置盘，从盘中读数据，然后做一些诸如导入公钥的动作。您可以在本文档中读到更多配置盘中数据格式的内容。

几个注意事项：

- 不要依赖配置盘中的EC2元数据，因为这些内容在新版本中可能会被移除。例如，不要依赖 `ec2` 文件中的文件。

- 在您创建可读取配置盘的镜像时，如果有 `openstack` 文件夹下有多个文件夹，您一定要选择用户支持的最高版本的API（以日期标注）。比如说，如果您的镜像支持2012-03-05，2012-08-05和2013-04-13版本，先尝试使用2013-04-13版本，如果该版本不存在，再尝试前面的版本。

启用和访问配置盘

1. 如果要启用配置盘，将 `--config-driver true` 参数传给 `nova boot` 命令即可。

在下面的例子里，我们启用了配置盘，将用户数据，两个文件，以及两个键/值元数据对传给了它，这些都可以在配置盘中获取到。

```
$ nova boot --config-drive true --image my-image-name --key-name my
  --flavor 1 --user-data ./my-user-data.txt myinstance \
  --file /etc/network/interfaces=/home/myuser/instance-interfaces \
  --file known_hosts=/home/myuser/.ssh/known_hosts \
  --meta role=webservers --meta essential=false
```

您也可以把Compute服务配置成每次都使用配置盘。在 `/etc/nova/nova.conf` 文件中配置如下条目：

```
force_config_drive=true
```

> 注意：如果某位用户将 `--config-drive true` 参数传递给了 `nova boot` 命令，连管理员也没法禁用。

1. 如果您的实例支持通过标签来访问磁盘，您可以以 `/dev/disk/by-label/configurationDriveVolumeLabel` 来挂载配置盘。在下面的例子中，配置盘的标签是 `config-2` ：

```
# mkdir -p /mnt/config
# mount /dev/disk/by-label/config-2 /mnt/config
```

> 注意： 为了对配置盘提供支持，您使用的CirrOS至少在0.3.1版本以上。 如果您的客户机不用 `udev` ，是不会有 `/dev/disk/by-label` 的。 您可以用 `blkid` 命令来查找配置盘对应的块设备。比如，如果您用CirrOS镜像，在 `m1.tiny` 的配置下启动了实例，配置盘应该是 `/dev/vdb` ：
>
> ```
> # blkid -t LABEL="config-2" -odevice
> /dev/vdb
> # mkdir -p /mnt/config
> # mount /dev/vdb /mnt/config
> ```

配置盘中的内容

下面这个例子中，配置盘中的内容如下：

```
ec2/2009-04-04/meta-data.json
ec2/2009-04-04/user-data
ec2/latest/meta-data.json
ec2/latest/user-data
openstack/2012-08-10/meta_data.json
openstack/2012-08-10/user_data
openstack/content
openstack/content/0000
openstack/content/0001
openstack/latest/meta_data.json
openstack/latest/user_data
```

配置盘中有哪些内容取决于您使用 `nova boot` 时传入了哪些参数。

OpenStack元数据格式

下面的内容展示了 `openstack/2012-08-10/meta_data.json` 和 `openstack/latest/meta_data.json` 文件。这两个文件是完全一样的。为了方便阅读，内容已经做过排版。

```
{
    "availability_zone": "nova",
    "files": [
        {
            "content_path": "/content/0000",
            "path": "/etc/network/interfaces"
        },
        {
            "content_path": "/content/0001",
            "path": "known_hosts"
        }
    ],
    "hostname": "test.novalocal",
    "launch_index": 0,
    "name": "test",
    "meta": {
        "role": "webservers",
        "essential": "false"
    },
    "public_keys": {
        "mykey": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQDBqUfVvCSez
    },
    "uuid": "83679162-1378-4288-a2d4-70e13ec132aa"
}
```

请注意您使用 `nova boot` 参数时配置的参数 `--file /etc/network/interfaces=/home/myuser/instance-interfaces` 产生的效果。这个文件的内容被保存在了 `openstack/content/0000` 文件中，路径信息指定为 `/etc/network/interface/`，保存在 `meta_data.json` 中。

EC2元数据格式

下面的内容展示了 `ec2/2009-04-04/meta_data.json` 和 `ec2/latest/meta_data.json` 文件。这两个文件是完全一样的。为了方便阅读，内容已经做过排版。

```
{
    "ami-id": "ami-00000001",
    "ami-launch-index": 0,
    "ami-manifest-path": "FIXME",
    "block-device-mapping": {
        "ami": "sda1",
        "ephemeral0": "sda2",
        "root": "/dev/sda1",
        "swap": "sda3"
    },
    "hostname": "test.novalocal",
    "instance-action": "none",
    "instance-id": "i-00000001",
    "instance-type": "m1.tiny",
    "kernel-id": "aki-00000002",
    "local-hostname": "test.novalocal",
    "local-ipv4": null,
    "placement": {
        "availability-zone": "nova"
    },
    "public-hostname": "test.novalocal",
    "public-ipv4": "",
    "public-keys": {
        "0": {
            "openssh-key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQD
        }
    },
    "ramdisk-id": "ari-00000003",
    "reservation-id": "r-7lfps8wj",
    "security-groups": [
        "default"
    ]
}
```


用户数据

`openstack/2012-08-10/user_data` ， `openstack/latest/user_data` ， `ec2/2009-04-04/user-data` 和 `ec2/latest/user-data` 这四个文件只有在您使用 `nova boot` 文件时传入了 `--user-data` 参数和用户数据文件时才会出现。

配置盘格式

默认的配置盘格式时 `ISO 9660` 。如果要显式地指定 `ISO 9660` 格式，请在 `/etc/nova/nova.conf` 文件中添加以下配置：

```
config_drive_format=iso9660
```

默认情况下，您只能将配置盘以硬盘的形式装载在实例上，而不能用光盘的形式。如果要以CD的形式装载，请在 `/etc/nova/nova.conf` 文件中添加以下配置：

```
config_drive_cdrom=true
```

为了提供对旧设备的支持，您还可以将配置盘设置为 `VFAT` 格式。您可能不会需要用到 `VFAT` ，因为 `ISO 9660` 已被绝大多数操作系统所支持。然而，要使用 `VFAT` 格式，请在 `/etc/nova/nova.conf` 文件中添加以下配置：

```
config_drive_format=vfat
```

如果您选择了 `VFAT` ，配置盘的大小将会是64 MB。

> 注意： 在当前的OpenStack版本中（Liberty），给用到 `config_drive` 的本地盘做热迁移的功能被禁用了，因为libvirt在复制只读盘时有bug。然而，如果我们使用 `VFAT` 格式作为 `config_drive` 的格式，热迁移的功能是可用的。

创建和管理网络

在对OpenStack网络进行操作前，请先设置如下环境变量：

```
export OS_USERNAME=admin
export OS_PASSWORD=password
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://localhost:5000/v2.0
```

创建网络

1. 列出neutron工具的扩展功能：

```
$ neutron ext-list -c alias -c name

+-----------------+--------------------------+
| alias           | name                     |
+-----------------+--------------------------+
| agent_scheduler | Agent Schedulers         |
| binding         | Port Binding             |
| quotas          | Quota management support |
| agent           | agent                    |
| provider        | Provider Network         |
| router          | Neutron L3 Router        |
| lbaas           | LoadBalancing service    |
| extraroute      | Neutron Extra Route      |
+-----------------+--------------------------+
```

1. 创建网络

```
$ neutron net-create net1

Created a new network:
+---------------------------+-------------------------------------
| Field                     | Value
+---------------------------+-------------------------------------
| admin_state_up            | True
| id                        | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| name                      | net1
| provider:network_type     | vlan
| provider:physical_network | physnet1
| provider:segmentation_id  | 1001
| router:external           | False
| shared                    | False
| status                    | ACTIVE
| subnets                   |
| tenant_id                 | 3671f46ec35e4bbca6ef92ab7975e463
+---------------------------+-------------------------------------
```

> 注意：
> 使用 `net-create` 命令创建网络时，返回信息中的一些项是只有管理员才能看到的。

1. 创建网络时，指定网络类型：

```
$ neutron net-create net2 --provider:network-type local

Created a new network:
+---------------------------+-------------------------------------
| Field                     | Value
+---------------------------+-------------------------------------
| admin_state_up            | True
| id                        | 524e26ea-fad4-4bb0-b504-1ad0dc770e7a
| name                      | net2
| provider:network_type     | local
| provider:physical_network |
| provider:segmentation_id  |
| router:external           | False
| shared                    | False
| status                    | ACTIVE
| subnets                   |
| tenant_id                 | 3671f46ec35e4bbca6ef92ab7975e463
+---------------------------+-------------------------------------
```

正如上面展示的一样，刚刚我们用到的那个 `--provider:network-type` 能用来创建一个 `local` 的provider network。

创建子网

```
$ neutron subnet-create net1 192.168.2.0/24 --name subnet1

Created a new subnet:
+------------------+-------------------------------------------------
| Field            | Value
+------------------+-------------------------------------------------
| allocation_pools | {"start": "192.168.2.2", "end": "192.168.2.254
| cidr             | 192.168.2.0/24
| dns_nameservers  |
| enable_dhcp      | True
| gateway_ip       | 192.168.2.1
| host_routes      |
| id               | 15a09f6c-87a5-4d14-b2cf-03d97cd4b456
| ip_version       | 4
| name             | subnet1
| network_id       | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| tenant_id        | 3671f46ec35e4bbca6ef92ab7975e463
+------------------+-------------------------------------------------
```

`subnet-create` 命令有如下几个位置固定的参数，还有可选参数：

- 这个子网从属的网络的名字或者ID 在本例中， `net1` 这个参数是位置固定的。
- 子网的CIDR 在本例中， `192.168.2.0/24` 也是一个位置固定的参数，它标记了要创建的子网的CIDR。
- 子网的名称 在本例中， `--name subnet1` 指定了要创建的子网名。

创建路由

1. 创建一个新路由

```
$ neutron router-create router1

Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 6e1f11ed-014b-4c16-8664-f4f615a3137a |
| name                  | router1                              |
| status                | ACTIVE                               |
| tenant_id             | 7b5970fbe7724bf9b74c245e66b92abf     |
+-----------------------+--------------------------------------+
```

记下返回的路由识别码，这个编码是唯一的，在稍后的步骤中我们将用到它。

1. 将路由器连接到外部的provider network。

```
$ neutron router-gateway-set ROUTER NETWORK
```

将这条命令中的 `ROUTER` 字段用刚刚的路由识别码代替，将 `NETWORK` 字段用唯一的外部provider network识别码代替。

1. 将该路由和子网相连。

```
$ neutron router-interface-add ROUTER SUBNET
```

将这条命令中的 `ROUTER` 字段用刚刚的路由识别码代替，将 `SUBNET` 字段用唯一的子网代码代替。

创建端口

1. 在指定的IP地址上创建一个端口

```
$ neutron port-create net1 --fixed-ip ip_address=192.168.2.40

Created a new port:
+-----------------------+-----------------------------------------
| Field                 | Value
+-----------------------+-----------------------------------------
| admin_state_up        | True
| binding:capabilities  | {"port_filter": false}
| binding:vif_type      | ovs
| device_id             |
| device_owner          |
| fixed_ips             | {"subnet_id": "15a09f6c-87a5-4d14-b2cf-03c
| id                    | f7a08fe4-e79e-4b67-bbb8-a5002455a493
| mac_address           | fa:16:3e:97:e0:fc
| name                  |
| network_id            | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| status                | DOWN
| tenant_id             | 3671f46ec35e4bbca6ef92ab7975e463
+-----------------------+-----------------------------------------
```

在前一个指令中， `net1` 参数表示的是网络名，该参数的位置要固定。 `--fixed-ip ip_address=192.168.2.40` 则是可选的，指定了该端口的绑定的IP地址是哪一个。

> 注意： 在创建端口时，您可以指定任意一个在子网中的未分配的IP地址，即便它不在您的云供应商提供的地址池内也可以。

1. 在不指定IP地址的情况下创建端口

```
$ neutron port-create net1

Created a new port:
+----------------------+------------------------------------------
| Field                | Value
+----------------------+------------------------------------------
| admin_state_up       | True
| binding:capabilities | {"port_filter": false}
| binding:vif_type     | ovs
| device_id            |
| device_owner         |
| fixed_ips            | {"subnet_id": "15a09f6c-87a5-4d14-b2cf-03d
| id                   | baf13412-2641-4183-9533-de8f5b91444c
| mac_address          | fa:16:3e:f6:ec:c7
| name                 |
| network_id           | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| status               | DOWN
| tenant_id            | 3671f46ec35e4bbca6ef92ab7975e463
+----------------------+------------------------------------------
```

> 注意： 如果您在使用 `neutron port-create` 指令时不指定IP地址，系统会自动分配给您一个IP地址。

1. 通过固定的IP地址来查询端口

```
$ neutron port-list --fixed-ips ip_address=192.168.2.2 \
  ip_address=192.168.2.40

+----------------+------+-------------------+---------------------
| id             | name | mac_address       | fixed_ips
+----------------+------+-------------------+---------------------
| baf13412-26... |      | fa:16:3e:f6:ec:c7 | {"subnet_id"... ..."i
| f7a08fe4-e7... |      | fa:16:3e:97:e0:fc | {"subnet_id"... ..."i
+----------------+------+-------------------+---------------------
```

其中， `--fixed-ips ip_address=192.168.2.2 ip_address=192.168.2.40` 是这个命令的unknown option。

如何查找**unknown option**：unknown option可以通过观察`create_xxx`或者`show_xxx`的指令来查找。比如，在用创建端口的命令时，我们能看见`fixed_ips`这一项，它便可以用作unknown option。

管理对象和容器

我感觉这章的内容我翻译了。但是似乎被抹掉了。也没保存。Crap。